The Inventory Routing Problem with time-dependent travel times

Wouter Lefever

Promotoren: prof. dr. El-Houssaine Aghezzaf, prof. Khaled Hadj-Hamou

Masterproef ingediend tot het behalen van de academische graad van
Master of Science in Industrial Engineering and Operations Research

Vakgroep Technische Bedrijfsvoering
Voorzitter: prof. dr. El-Houssaine Aghezzaf
Faculteit Ingenieurswetenschappen en Architectuur
Academiejaar 2013-2014

UNIVERSITEIT
GENT

Grenoble, June 2014

The supervisors	The author
Prof. El-Houssaine Aghezzaf, Prof. Khaled Hadj-Hamou	Wouter Lefever

Acknowledgements

I would like to express my special thanks of gratitude to both my supervisors, Professor El-Houssaine Aghezzaf and Professor Khaled Hadj-Hamou. Professor Aghezzaf, when I decided to participate in the ERASMUS exchange program, your support and guidance were most valuable to me. I belief that it requires faith to have one of your students writing his master dissertation abroad and I would like to thank you for the faith you had in me and for your support all throughout my master dissertation. Professor Hadj-Hamou, it requires courage to send a student abroad, but it requires as much courage to accept a foreign student. I would like to thank you for the opportunity to work under your supervision on this master dissertation. Working with you was both an honour and a pleasure.

I am also thankful for the unique scientific environment at the G-SCOP laboratory and all the professors, assistants, interns,... who helped in creating this stimulating environment. Working here on my master dissertation has been a really valuable experience for me. I am particularly grateful for the assistance given by the technical staff whenever I encountered computer-related problems.

This list of acknowledgements would be incomplete without my sincere appreciation for all the people I met during my stay in Grenoble. Thanks to all my friends here who have made this year a great experience.

Last but not least, I would like to thank my parents, Bart and Katrien, my sisters, Anneleen and Marlies, and my brother Bram for their unceasing encouragement and support.

Notation

Sets

S	Set of all Scenarios
A	Set of all arcs in the graph $G = (V, A)$
V	Set of all vertices in the graph $G = (V, A)$
$V^{'}$	Set of all clients in the graph $G = (V, A)$
T	Set of all periods

Parameters

r^t	Amount of goods produced at the supplier in period t
d_i^t	Demand of client i in period t
Q	Vehicle capacity
C_0	Inventory Capacity of the supplier
C_i	Inventory Capacity of client i
h_i	Holding costs per unit per period for client i
c_{ij}	Transportation costs between location i and location j
t_{ij}	Travel time between location i and location j
$TimeLimit$	Time in which the tour has to be completed to avoid paying a $Penalty$
$Penalty$	Penalty that thas to be paid when tour takes longer to complete than $TimeLimit$

Variables

I_0^t Inventory of the supplier in period t

I_i^t Inventory of client i in period t

q_i^t Amount of goods delivered at client i in period t

x_{ij}^t Binary decision variable

$$x_{ij}^t = \begin{cases} 1 & \text{if road (i,j) is used in period t} \\ 0 & \text{otherwise} \end{cases}$$

y_i^t Binary decision variable

$$y_i^t = \begin{cases} 1 & \text{if actor i is visited in period t} \\ 0 & \text{otherwise} \end{cases}$$

p_t Binary decision variable

$$p_t = \begin{cases} 1 & \text{if tour in period t is bigger than the limit } \mathit{TimeLimit} \\ 0 & \text{otherwise} \end{cases}$$

Acronyms

GDP Gross Domestic Product

VMI Vendor-Managed Inventory

IRP Inventory Routing Problem

IRPVTT Inventory Routing Problem with Variable Travel Times

MIP Mixed Integer Program

VRP Vehicle Routing Problem

TSP Travelling Salesman Problem

TSPSC TSP with Stochastic Customers

TSPST TSP with Stochastic Travel Times

VRPSD VRP with Stochastic Demand

VRPST VRP with Stochastic Travel Times

ML Maximum Level

OU Order-Up-To Level

SOCP Second-Order Cone Program

Table of Contents

List of Figures

List of Tables

Summary

In the current business context, companies are confronted with fierce competition. Logistics is often a key component in gaining advantage over the direct competitors. Vendor-Managed Inventory(VMI) systems are a relatively recent strategy in logistics. In these systems the supplier takes full responsibility for the inventory of his clients. This business model is often regarded as a win-win situation for both supplier and clients. The client doesn't have to put effort into monotoring his inventory and passing orders to the supplier. The supplier can combine orders easier and thus save in distribution costs.

The mathematical model behind VMI system is the Inventory-Routing Problem(IRP). This problem minimizes holding and transportation costs, while making inventory management and routing decisions over several periods. In this model travel times are assumed to be predictable, while in reality this is not the case. To overcome this gap, this master dissertation presents methods to address the IRP with variable travel times(IRPVTT).

In the development of the solution methods the focus lies on two criteria: execution time and accuracy. Execution time is a necessary criterion, since we are dealing with NP-problems and the introduction of variability makes the problem even harder. Accuracy is a good criterion to evaluate the quality of the solution.

Four different solution methods are presented in this master dissertation. The first two solution methods try to identify interesting solutions over the full range of variability. Afterwards Monte Carlo simulation is applied to distinguish the best among the identified solutions. The third method solves the problem for a set of scenarios. Based on the results new scenarios are introduced in the scenario set. Finally, the fourth method also solves the problem for a set of scenarios, but in the fourth method the scenarios are chosen randomly. Opposed to the third method, there is thus no interaction between the solution of the problem and the selection of the next scenario.

For these four methods improvements are suggested to improve their performance in execution time and accuracy. For the first two methods this includes a proper stopping criterion. For the last two methods a wide range of improvements(heuristics) is possible to improve their

execution time: fixation of variables, removing of scenarios out of the set, stopping criterion...

The master dissertation is structured as follows. After an introduction and motivation for this study, we present the literature review of the related themes, followed by three chapters in which solution methods are presented, improved and evaluated. Conclusions and directions for future work are presented in the last chapter.

Keywords: Inventory-routing problem; variability; Mixed Integer Program; heuristic; execution time; accuracy.

Chapter 1

Introduction

In the current business environment logistics is one of the key elements to gain a competitive advantage over the direct competitors. Improvements in logistics can translate into significant savings for a company. The exact percentage of cost savings depends on the structure of the firm and the economical sector it operates in, but percentages up to 30% have been estimated in scientific studies(Said et al.[25]). Not only micro-economically, but also macro-economically logistics play an important role. Recent research(Shepherd[26]) shows that in 2008 the logistics sector in France, depending on the definition of logistics, accounts between 4,41% and 12,79% of the Gross Domestic Product(GDP). In Belgium these numbers even raise to 5,89% and 17,04% of the GDP. Research in this domain is thus of major importance to the community.

Recently, Vendor-Managed Inventory (VMI) systems have emerged as an important component in the logistic strategies. In this business model the supplier takes responsibility for the management of the client's inventory until the moment of use. Both supplier and client benefit from this practice. The client doesn't have to put effort into monitoring and maintaining his inventory. The supplier can combine deliveries to multiple clients more effectively and thus save distribution costs. The decisions that the supplier has to take in this practice are when to deliver, which quantity and which tour to follow.

The Mixed Integer Program(MIP) that models VMI systems is called Inventory-Routing Problem(IRP). These models minimize inventory costs for both the supplier and the clients and the transportation costs between supplier and clients. In the recent years a lot of research has been done on IRP. Coelho et al.[11] provides an extensive overview of different types of IRP and solution methods.

However, most of these models assume predictable travel times(and associated costs). In real life this is usually not the case. Many factors influence the travel time between two locations. These factors can be both endogenous(for example capacity and occupation of the road) or

exogenous(weather conditions or accidents). Approaching travel times by one parameter, the average, may thus be unsuited. To overcome this gap between theoretical models and reality our aim is to include variability on travel times in the IRP model. We will do this by modelling the travel time t_{ij} between two locations i and j as a symmetric and bounded random variable $\tilde{t}_{ij}$ that takes values in $\left[t_{ij} - \hat{t}_{ij}, t_{ij} + \hat{t}_{ij}\right]$. t_{ij} and $\hat{t}_{ij}$ being respectively the nominal value and the maximum deviation of the nominal value. When the problem is solved over multiple periods, the travel times in one period are independent of the travel times in the next period. There is thus no additional information being revealed over time. In our interpretation of the problem there is no difference between time-dependent travel times or variable travel times. By approaching the travel times in this manner we hope to include the influence of endogenous factors. Exogenous factors are left out of our approach because of their irregular nature.

As a result of the introduced variability the optimal solution of the nominal problem may produce very different, sub-optimal, results under various circumstances. In this context we would like to cite the conclusion of Ben-Tal and Nemirovski[7] of their case study on linear optimization problems form the Net Lib library:

"In real-world applications of Linear Programming one cannot ignore the possibility that a small uncertainty in the data (intrinsic for most real-world LP programs) can make the usual optimal solution of the problem completely meaningless from a practical viewpoint."

The need arises thus to have models that can cope with this uncertainty. There are two factors that complicate the development of such models. First the strict sense of optimality that exists in linear programming without uncertainty disappears when uncertainty is taken into account. Because of the continuous nature of travel time, it is impossible to verify a solution for all possible values of $\tilde{t}_{ij}$ and thus prove its optimality. To distinguish the better between several solutions Monte Carlo simulation can be used, but it doesn't provide us with a proof of optimality, only an indication. Furthermore, the uncertainty appearing in the travel times will propagate via the costs of the travel times to the total cost. Now optimality may be defined in different ways: the solution performing best on average, the solution having the least variability in its total cost, ... We will use the first definition of optimality in our study, but we do not exlude the usefulness of other definitions. The second complicating factor is the NP-completeness of the IRP. Since the IRP with variable travel times(IRPVTT) envelops the nominal problem, the former will also be NP-complete. Therefore, large instances might require exponential calculation time to solve to optimality.

These complications make that we will focus on two main aspects in the development of our models: accuracy and execution time. Accuracy will evaluate how likely a model is to find the, to our knowledge, optimal solution. Focussing on execution time will avoid developping

models that take enormous amounts of time to come up with a solution. Notice that good performance in both criterions is needed for any model to be used in practice. Excellence in execution time is meaningless when the model fails to find good solutions and excellence in accuracy loses it attractiveness when it takes infinite time to solve a problem.

Our main goal will thus be to find a solution within reasonable time that is optimal in the sense we use. An additional goal is to estimate the distribution of the total cost corresponding with our solution.

The remainder of this master dissertation is organized as follows. In chapter 2 we present the literare review. We distinguish three parts in the literature review. In the first part we will present the basic IRP model that will be used throughout the rest of the study. This model was discribed as a standard version of the IRP model in Coelho et al.[11]. In the second part we review the literature on methods that have been used to address variability or uncertainty in the IRP. Because the IRP is heavily related to the Vehicle Routing Problem(VRP) and Travelling Salesman Problem(TSP), we will include research on variability in VRP and TSP also in our review. In the last part of the lierature review we present a brief summary on the main approaches towards robust optimisation. We will integrate robust optimisation in one of our models in chapter 3.

In chapter 3 we present the 4 different methods to approach the problem. The first two methods will in a first stage try to identify interesting solutions for the IRPVTT. In a second stage Monte Carlo simulation is used to distinguish the best among the solutions found. The third and fourth method differ fundamentally of the first two methods. In the last two methods the problem will be solved for a set of scenarios S which will provide us directly with a solution for the IRPVTT. In these methods there is thus no need for Monte Carlo simulation afterwards.

We will develop these methods further in chapter 4. As explained in the previous paragraphs the main focus will be on improvements in accuracy and execution time. At the end of each section we will comment on some of the advantages and drawbacks of the presented method.

In chapter 5 we will compare the methods by use of simulation under different configurations and discuss our findings from the experiments.

In chapter 6 we discuss how the cost corresponding with a solution can be estimated. This discussion only holds for the last two methods, since the first two methods apply Monte Carlo simulation to evaluate different solutions.

Finally, in chapter 7 we present the main conclusions of our research and point out directions for future work.

Chapter 2

Inventory Routing problem: A short literature review

In this chapter we present a standard version of the IRP put forward in Coelho et al.[11]. In the second part we review which methods have been used in the past to address variability in the IRP and similar problems. Finally we review briefly the existing literature on robust optimisation.

2.1 Inventory Routing Problem

The IRP calls for the determination of the optimal set of routes to be performed by a fleet of vehicles to serve a given set of customers taking into consideration both transportation and inventory costs. The problem was first described in a seminal paper of Bell et al. (1983)[4]. Following this paper, a wide range of models and algorithms were proposed to find optimal and approximate solutions for the different versions of the IRP. A compresensive literature overview can be found in Coelho et al.[11].

Even though there exist many variants on the IRP, most of them start from the same basic structure. In this basic version there's one supplier who delivers to multiple customers. The delivered goods are homogeneous in any aspect and the delivering happens over a finite time horizon. Furthermore, the lead time of all manufacturing operations is assumed to be 0. Regarding the inventory policy, a maximum level (ML) policy is used, meaning that delivered quantities have to respect the inventory capacity of every customer. A stricter inventory policy is order-up-to level (OU) where the delivered quantities have to fill the inventory to its maximum capacity. All demand has to be delivered and no back-orders are allowed. This imposes a non-negativity condition on our inventory. Concerning the fleet only one vehicle is used with limited capacity to deliver the goods.

To model this problem mathematically we will start with defining a graph $G = (V, A)$ in which the collection of vertices $V = \{0, ..., n\}$ represent the actors,the supplier and the customers, and the arcs $A = \{(i, j) : i, j \in V, i \neq j\}$ represent all routes between the actors. Vertex 0 will represent the supplier so that the collection $V' = V/\{0\}$ consists of all customers.

Since the problem is solved over multiple time periods, we have to introduce a time dimension $t \in T = \{1..p\}$ in which p is the length of the planning horizon. Every time period t a certain amount of goods is produced r^t at the supplier. These goods are added to the inventory of the supplier I_0^t. When the supplier delivers to a customer i he will take a quantity q_i^t of his own inventory and add this quantity to the inventory of the customer I_i^t. The customer will use this inventory to fulfill its demand d_i^t.

The initial parameters for the model are the capicity of the vehicle Q, the initial inventories of the supplier and the customers I_i^0, $i \in V$, and the inventory capacities of the supplier and the customers C_i, $i \in V$. Furthermore, the holding costs for the supplier and the customers h_i are given as well as the transportation costs between supplier and customers c_{ij}.

The modelling of the IRP as an MIP happens by use of four variables: x_{ij}^t, y_i^t, I_i^t and q_i^t. These variables are defined in the following way:

$$x_{ij}^t = \begin{cases} 1 & \text{if road (i,j) is used in period t} \\ 0 & \text{otherwise} \end{cases}$$

$$y_i^t = \begin{cases} 1 & \text{if actor i is visited in period t} \\ 0 & \text{otherwise} \end{cases}$$

I_i^t : Inventory level of actor i in period t

q_i^t : Amount of goods delivered at client i in period t

Using the sets, parameters and variables described above, the MIP model for the basic version of the IRP can be written as:

$$minimize \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{i \in V} \sum_{j \in V, i<j} \sum_{t \in T} c_{ij} x_{ij}^t \tag{1}$$

$$subject\ to\ I_0^t = I_0^{t-1} + r^t - \sum_{i \in V'} q_i^t \qquad t \in T \tag{2}$$

$$I_0^t \geq 0 \qquad t \in T \tag{3}$$

$$I_i^t = I_i^{t-1} + q_i^t - d_i^t \qquad t \in T,\ i \in V' \tag{4}$$

$$I_i^t \geq 0 \qquad t \in T,\ i \in V' \tag{5}$$

$$I_i^t \leq C_i \qquad t \in T,\ i \in V \tag{6}$$

$$q_i^t \leq C_i - I_i^{t-1} \qquad t \in T,\ i \in V' \tag{7}$$

$$q_i^t \leq C_i y_i^t \qquad t \in T,\ i \in V' \tag{8}$$

$$\sum_{i \in V'} q_i^t \leq Q y_0^t \qquad t \in T \tag{9}$$

$$\sum_{j \in V, i<j} x_{ij}^t + \sum_{j \in V, i>j} x_{ji}^t = 2 y_i^t \qquad t \in T,\ i \in V \tag{10}$$

$$\sum_{i \in S} \sum_{j \in S, i \leq j} x_{ji}^t \leq \sum_{i \in S} y_i^t - y_m^t \qquad S \subset V',\ t \in T,\ m \in S \tag{11}$$

$$x_{0j}^t \in \{0, 1, 2\} \qquad j \in V',\ t \in T \tag{12}$$

$$x_{ij}^t \in \{0, 1\} \qquad i, j \in V',\ t \in T \tag{13}$$

$$y_i^t \in \{0, 1\} \qquad i \in V,\ t \in T \tag{14}$$

$$q_i^t \geq 0 \qquad i \in V',\ t \in T \tag{15}$$

In this formulation the objective function (1) minimizes the holding costs of all actors and transportation costs between all actors over all periods. Constraints (2) model the inventory of the supplier over the different periods and constraints (3) guarantee the non-negativity condition on the inventory of the supplier. Likewise constraints (4) model the inventory of the customers and constraints (5) guarantee the non-negativity condition. Constraints (6) ensure that no inventory will exceed its maximum capacity. Constraints (7) limit the delivered quantity based on the inventory in the last period and the capacity of the inventory, while constraints (8) ensure that quantities can only be delivered if the customer is visited that period. Constraints (9) model the vehicle capacity. To model the tours properly we need constraints (10), degree constraints, and constraints (11) to eliminate subtours. Finally constraints (12)-(15) limit the domain of the variables. Notice that routes leaving from the supplier can take the value 2 which represents the presence of a 2-tour in the solution.

As can be seen travel times t_{ij} don't appear directly in the MIP, given above. However, we can assume that the transportation costs c_{ij} are in some way related to the travel times. For

example costs of the fuel and the wage of the driver are linked to the travel time. Due to this relation, variability in travel times will propagate to the transportation costs c_{ij} in the objective function. This link between transportation costs and travel times will further be explicited in section 3.1.

In the next sections we will often refer to different instances of the IRP. All instances were downloaded from the following site:

http://www.leandro-coelho.com/instances/thesis/exact_irp/

2.2 Methods addressing variability in VRP and IRP

After the presentation of the standard model in the previous section, we will review relevant literature on variability in TSP, VRP and IRP in this section. Compared to their deterministic counterparts there is little research conducted on TSP, VRP and IRP with stochastic elements. The main reason for this lack of research is the huge increase in complexity that the inclusion of stochastic elements brings to these already difficult problems. Variability can be introduced in a number of ways. Examples are TSP with Stochastic Customers(TSPSC), TSP with Stochastic Travel Times(TSPST), VRP with Stochastic Demand(VRPSD) and many more.

An article that gives an excellent overview of the research done until 1996 in stochastic vehicle routing is Gendreau et al.[16]. An similar discussion on VRP with Stochastic Travel Times(VRPST) and VRPSD is found in Hadjiconstantinou and Roberts[19]. We are mainly interested in stochastic travel times, but we will see that methods addressing stochastic demand are also of use for our study. In their study Gendreau et al.[16] state that the objective for VRPST or TSPST is often to find an a priori solution that maximises the probability of completing any tour within a given deadline. They remark that for this problem no exact methods under the form of a mathematical model have ever been presented. Kao[20] presented two heuristic methods to solve this problem. One based on dynamic programming and one based on implicit enumeration. Based on this research Sniedovich[27] demonstrated that the property of monotonicity is required for the last heuristic to obtain optimal solutions. To overcome this problem Carrayway et al.[9] Presented a more general version of the dynamic programming method.

When the extension is made to the multiple vehicle TSPST (m-TSPST), which is interchangeable with the VRP with Stochastic Travel Times(VRPST), work of Lambert et al.[21] is well worth noting. They developed a heuristic solution algorithm for the m-TSPST and found cost-effective cash-collection routes through bank branches. In these routes the amount of

cash collected was limited by an insurance company and late arrival was punished by loss of interest. For the VRPST we remark the study of Laporte et al.[22]. They changed the model of the VRPST which allowed them to consider an alternative objective in which the penalty for late arrival is proportional to the length of the delay.

Next to the methods addressing variability in travel times, we also review the literature in which the demand is stochastic. First work in this domain was done by Tillman[33] who proposed an algorithm, based on Clarke and Wright's[10] savings procedure, to include stochastic demand. Other heuristic methods as simulated annealing by Teodorović and Pavković[32], tabu search by Gendreau et al.[17] and an route-first, cluster-second approach by Teodorović, Krcmar-Nozić and Pavković[31] were later developed for the same problem.

Next to the heuristic methods also mathematical models were presented. Golden and Stewart[18] proposed as first to apply stochastic programming to the VRPSD. The same authors extended their study in Stewart and Golden[30] where the chance constrainted case, customers are served according to a given probability, and penalty function case, where each customer is served with the inclusion of a possible recourse cost, were included. On their work other stochastic formulations were presented in Dror and Trudeau[15], Dror et al.[14], Bastian and Kan[3],Dror [12], Dror et al.[13] and Popović[23]. However non of these models resulted in exact solutions. Finally we mention work of Roberts and Hadjiconstantinou[24] who presented a new method that successfully solves VRPSDs of medium size.

When we review the literature on the Stochastic IRP (SIRP), we remark two interesting studies. First there is Aghezzaf[1] who allows variability in both travel times and customer demands. Robust optimization is used to solve the problem and come up with a distribution plan. Afterwards Monte Carlo simulation is applied to find better values for some parameters. Second there is Solyali et al.[28] They apply a robust optimization approach, developed by Bertsimas and Sim[8], to solve the IRP with stochastic demand. In their study they state:

"The robust solution thus obtained provide immunization against uncertainty with a slight increase in total cost compared to the nominal case, especially when the average daily demand over vehicle capacity ratio is low, whereas the price of robustness is larger when the average daily demand over vehicle capacity is high."

Because we are facing similar challenges as in Solyali et al.[28] and their positive results, robust optimization might be an interesting approach towards solving our problem. Therefore, we will review three breakthroughs in this domain in the next section.

2.3 Robust Optimization

In robust optimization the goal is to optimize against the worst instances that result from uncertainty. Consider the following nominal linear problem:

$$\begin{aligned} &maximize\ cx \\ &subject\ to\ Ax \leq b \\ &\qquad\qquad l \leq x \leq u \end{aligned}$$

For every row i of the matrix A, we will note J_i as the set of uncertain coefficients in this row. Every uncertain coefficient $\tilde{a}_{ij}$, $j \in J_i$, will take random values in a symmetric interval around the nominal value : $[a_{ij} - \hat{a}_{ij}, a_{ij} + \hat{a}_{ij}]$. To solve the problem under uncertain coefficients, we will evaluate three approaches. Note that in the linear program above uncertainty is only introduced on the coefficients a_{ij} in the constraints. The extension to variability on the coefficients in the objective function c_j is always possible.

2.3.1 Worst-case approach

Given this general problem Soyster[29] proposed a linear optimization model which constructs a solution that is feasible for the whole range of uncertainty. The model they propose will thus solve the worst-case scenario:

$$\begin{aligned} &maximize\ cx \\ &subject\ to \sum_j a_{ij}x_j + \sum_{j\ \in\ J_i} \hat{a}_{ij}y_j \leq b_i && \forall i \\ &\qquad -y_j \leq x_j \leq y_j && \forall j \\ &\qquad l \leq x \leq u \\ &\qquad y \geq 0 \end{aligned}$$

At optimality y_j will equal $|x_j^*|$, resulting in

$$subject\ to \sum_j a_{ij}x_j + \sum_{j\ \in\ J_i} \hat{a}_{ij}|x_j^*| \leq b_i \qquad \forall i$$

This will result in a solution x^* that is feasible for the whole range of uncertainty. However, protecting against the whole range of uncertainty might finally give an objective function

value that is much worse than the objective function value of the solution of the nominal linear optimization problem. In other words the conservatism of protecting againt the whole range of uncertainty may lead to an enormous decrease of the solution's value. Therefore, in the sections 2.3.2 and 2.3.3 we will consider other models that don't protect against the full range of uncertainty.

2.3.2 Uncertainty set

To cope with the problem of overconservatism Ben-Tal and Nemirovski[5][6][7] proposed a different robust problem. Their approach is based on letting the uncertain variable take values within an uncertainty set:

$$
\begin{aligned}
&\textit{maximize } cx \\
&\textit{subject to } \sum_j a_{ij}x_j + \sum_{j \in J_i} \hat{a}_{ij}y_j + \Omega_i \sqrt{\sum_{j \in J_i} \hat{a}_{ij}^2 z_{ij}^2} \leq b && \forall i \\
&\quad -y_j \leq x_j - z_{ij} \leq y_j && \forall i, j \in J_i \\
&\quad l \leq x \leq u \\
&\quad y \geq 0
\end{aligned}
$$

They showed in their work that in this model every constraint i is violated with a probability of at most $exp(-\Omega_i^2/2)$. This approach is less conservative than Soyser's approach. However, it changes the nominal linear problem into a Second-Order Cone Program(SOCP). This SOCP is much harder to solve than the MIP we started from. Bringing in mind that the IRP problem on its own is already a hard combinatorial problem, this approach may not be suited to apply to our problem.

2.3.3 Worst variable approach

To deal with the problem of overconservatism on one hand and hard non-linear models on the other Bertsimas and Sim[8] propose the following formulation:

$$
\begin{aligned}
&\textit{maximize } cx \\
&\textit{subject to } \sum_j a_{ij}x_j + \max_{S_i \cup \{t_i\} | S_i \subseteq J_i, |S_i| \leq \lfloor \Gamma_i \rfloor, t_i \in J_i n S_i} \left\{ \sum_{j \in S_i} \hat{a}_{ij}y_j + (\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{it_i}y_t \right\} \leq b && \forall i \\
&\quad -y_j \leq x_j \leq y_j && \forall j \in J_i \\
&\quad l \leq x \leq u \\
&\quad y \geq 0
\end{aligned}
$$

This model introduces a parameter Γ_i which takes values in $[0, |J_i|]$ and stands for the level of conservatism. The model will fully protect against $\lfloor \Gamma_i \rfloor$ uncertain coefficients and partly, only a variation of $(\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{it}$, against one coefficient a_{it}

If Γ_i is chosen integer, than increasing Γ_i with one, will result in protection against one more coefficient. Fixing Γ_i on zero results in the nominal problem. Fixing Γ_i on J_i results in Soyster's method.

Note that this model is still non-linear. Therefore, the authors used dual variables to linearlize the model. First the maximization problem that appears in the constraints is reformulated. For a given vector x^*, the subproblem becomes:

$$\beta_i(x^*) = \max_{S_i \cup \{t_i\} | S_i \subseteq J_i, |S_i| \leq \lfloor \Gamma_i \rfloor, t_i \in J_i n S_i} \left\{ \sum_{j \in S_i} \hat{a}_{ij} |x_j^*| + (\Gamma_i - \lfloor \Gamma_i \rfloor) \hat{a}_{it_i} |x_j^*| \right\}$$

This equals to :

$$\begin{aligned} \beta_i(x^*) = & maximize \sum_{j \in J_i} \hat{a}_{ij} |x_j^*| z_{ij} \\ & subject\ to \sum_{j \in J_i} z_{ij} \leq \Gamma_i \\ & 0 \leq z_{ij} \leq 1 \qquad \forall i \in J_i \end{aligned}$$

The optimal solution of this subproblem consists of $\lfloor \Gamma_i \rfloor$ variables at 1 and one variable at $\Gamma_i - \lfloor \Gamma_i \rfloor$. The solution corresponds to the selection of a subset $S_i \cup \{t_i\} | S_i \subseteq J_i, |S_i| \leq \lfloor \Gamma_i \rfloor, t_i \in J_i n S_i$ with cost function $\sum_{j \in S_i} \hat{a}_{ij} |x_j^*| + (\Gamma_i - \lfloor \Gamma_i \rfloor) \hat{a}_{it_i} |x_j^*|$.

Now we consider the dual of the previous subproblem:

$$\begin{aligned} & minimize \sum_{j \in J_i} p_{ij} + \Gamma_i z_i \\ & subject\ to\ z_i + p_{ij} \geq \hat{a}_{ij} |x_j^*| \quad \forall i, j \in J_i \\ & p_{ij} \geq 0 \qquad \forall j \in J_i \\ & z_i \geq 0 \qquad \forall i \end{aligned}$$

Substituting this subproblem into our original problem, we obtain the following linear optimization model. Note that $|x_j^*|$ is linearized by using a variable y_j and adding the constraint $-y_j \leq x_j \leq y_j$.

$$
\begin{aligned}
&\textit{maximize } cx \\
&\textit{subject to } \sum_j a_{ij}x_j + z_i\Gamma_i + \sum_{j \in J_i} p_{ij} \leq b_i && \forall i \\
&\qquad z_i + p_{ij} \geq \hat{a}_{ij}y_j && \forall i, j \in J_i \\
&\qquad -y_j \leq x_j \leq y_j && \forall j \\
&\qquad l_j \leq x_j \leq u_j && \forall j \\
&\qquad p_{ij} \geq 0 && \forall i, j \in J_i \\
&\qquad y_j \geq 0 && \forall j \\
&\qquad z_j \geq 0 && \forall j
\end{aligned}
$$

This method solves the two problems appearing in the approaches of Soyster and Ben-Tal-Nemirovski. The final formulation is linear and thus easier to solve than the SOCP of Ben-Tal and Nemirovski's approach. Introducting the parameter Γ_i, the model allows to control the level of conservatism contrary to Soysters's approach in which the problem is solved for the worst case instance.

Finally note that in formulation above uncertainty is only introduced on the parameters in the constraints. By introducing one extra variable and modelling the objective function as a constraint we are able to use the same approach for uncertain parameters that may appear in the objective function.

2.3.4 Conclusion

Given the fact that the IRP is already a hard problem, we are looking for a relatively simple method that allows us to find a robust solution. The approach of Ben-Tal and Nemirovski results in a SOCP, which is much harder to solve than our original MIP. Therefore, this approach is not really suited for our purpose. Soyster's approach is very unflexible and the overconservatism may lead to a great decrease of the quality of the solution. Since the approach of Bertsimas and Sim is more flexible, we have chosen to use this approach.

Chapter 3

Overview of proposed approaches

In this section we will present an overview of the solution methods we propose to solve the IRPVTT, but first we will adapt the model in order to make it more sensitive to changes in the travel times.

3.1 Extended model

Before analyzing the influence of variability introduced on one or more parameters in a model, it is necessary to first ensure that the model and the set of instances it solves are in fact sensitive to variations on these parameters. If the model is insensitive to variations in some parameters, no meaningful analysis can be conducted.

Keeping this in mind we will take a closer look at the MIP model of the IRP. As told in section 2.1, travel times t_{ij} are only represented indirectly in the transportation cost c_{ij} associated with a certain route(assuming that there exists a relation between the travel time and the cost of a certain route).

Since travel times only appear indirectly in the objective function, we will look at the portion of the total cost they represent. In table 3.1 this portion is calculated for the optimal solution of a series of instances in which the number of clients is varying. We notice that increasing the number of clients, decreases the importance of the transportation costs (and thus the sensitivity to the travel times). Variations on c_{ij} might have little influence when the transportation costs are dominated by the holding costss.

To cope with this problem we propose to extend the model. We will add a limit $TimeLimit$ in which a tour has to be completed. Violating this limit results in paying a penalty $Penalty$ which is added to the cost of the solution. In reality the time limit of the tour might for

Number of clients	5	10	15	20	25	30
Total costs	2108,34	4510,61	5589,7	6859,02	8227,86	12066,86
Transportation costs	1141,0	1440,0	1750,0	1760,0	2245,0	2471,0
Holding costs	967,34	3070,61	3839,7	5099,02	5982,86	9595,86
Percentage transportation costs	54,12%	31,92%	31,31%	25,66%	27,29%	20,48%

Table 3.1: Transportation costs as percentage of total cost in function of the number of clients

example represent the time a truck driver is allowed to drive without a mandatory break to rest. We can think of the penalty as all the extra costs induced by violating the time limit.

To add these changes to the MIP we will explicit the assumption that time and cost of a route are related: $c_{ij} = f(t_{ij})$. For simplicity we will assume that travel times and transportation costs are linked in the most simple linear way: $c_{ij} = t_{ij}$. Furthermore we will introduce a new variable to decide if a penalty has to be payed:

$$P_t = \begin{cases} 1 & \text{if tour in period t is bigger than the limit } \mathit{TimeLimit} \\ 0 & \text{otherwise} \end{cases}$$

The model now becomes:

$$min \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{i \in V} \sum_{j \in V} \sum_{t \in T} c_{ij} x_{ij}^t + \sum_{t \in T} PenaltyP_t \tag{1}$$

$$subject\ to\ (2) - (15)$$

$$\sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}^t \leq TimeLimit + MP_t \ \forall t \in T \tag{16}$$

$$P_t \in \{0, 1\} \ \forall t \in T \tag{17}$$

Note that constraint 16 is thus actually a time constraint. Only by expliciting the link between time and cost, we can use the cost coefficients also in this constraint. In this constraint M stands for a large value. Having P_t equal 1, will add this large value to the right-hand side of the constraint, making sure that it is respected.

The propagation of variability makes that $\tilde{c}_{ij}$ is now a symmetric and bounded random variable that takes values in $[c_{ij} - \hat{c}_{ij}, c_{ij} + \hat{c}_{ij}]$. c_{ij} and $\hat{c}_{ij}$ being respectively the nominal value and the maximum deviation of the nominal value. By varying the values of $TimeLimit$ and $Penalty$ we can now cope with cases where the transportation costs are heavily dominated by holding costs. Notice that either fixing $TimeLimit$ on ∞ or $Penalty$ on 0 results in the original model.

3.2 Solution methods

We will now present one by one the four approaches: the naive approach, the robust approach, the two-phase approach and the scenario approach.

3.2.1 Naive approach

A first approach to finding an optimal solution for the IRPVTT is to find different solutions for the nominal problem and evaluate the costs of these solutions when subject to variations in travel times.

We will first concentrate on finding different, and preferably interesting, solutions for the nominal problem. Whether a solution is interesting or not will heavily depend on the $TimeLimit$ we impose on our tours. When we solve the nominal problem and find the longest tour associated with its solution, then the difference between this longest tour, measured in time, and the $TimeLimit$ will determine if this solution is also interesting for variable IRP. When the value $TimeLimit$ is much higher than the longest tour, the nominal solution will be a very interesting candidate for the IRPVTT, since it is very unlikely that a penalty will have to be paid. The higher the difference between $TimeLimit$ and the longest tour, the more the variable IRPVTT resembles the nominal problem.

However, the smaller the difference between $TimeLimit$ and the longest tour becomes, the higher the probability that a penalty has to be paid becomes. This will make other solutions with shorter tours more interesting. In what follows we will fix $TimeLimit$ on the value of the longest tour of the nominal solution, unless explicitly stated otherwise.

Having this in mind, we can already label all solutions with larger tours than the longest tour of the optimal nominal solution as uninteresting. Their mean nominal costs are higher and their probability of having to pay a penalty is higher. It will thus be sufficient to look at the solutions with smaller tours than the longest tour of the optimal nominal solution. We can identify those solutions with the following procedure:

Algorithm 1 Identifying interesting solutions

Start with the nominal problem P ▷ Step 1
while Problem P has a feasible solution **do** ▷ Step 2
    Identify the longest tour of the optimal solution T_L
    Add constraints to problem P so that every tour is strictly smaller than T_L
end while

After identifying the interesting solutions we can use Monte Carlo simulation to estimate the

cost distribution of each solution. As said in the introduction, we will use the mean value as criterion to evaluate different solutions.

For the instance 'abs1n15', an instance with 15 clients, the solutions are depicted in figure 3.1. The longest tour of the optimal nominal solution is 1686 and we assumed a penalty of 500,0 for violating $TimeLimit$. We can see that the cost for solutions of the nominal problem rises if we enfore shorter tours(line in red). The cost of the variable IRP after Monte Carlo simulation(line in green) is different from the cost line of the nominal problem. We can clearly see a higher cost for solutions with longer tours. This higher cost is due to the higher probability of violating the $TimeLimit$. For solution with shorter tours the variable IRP cost and nominal IRP cost are almost the same, since there's very little chance of having to pay the penalty. The optimal solution of the variable IRP has a maximum tour length of 1498.

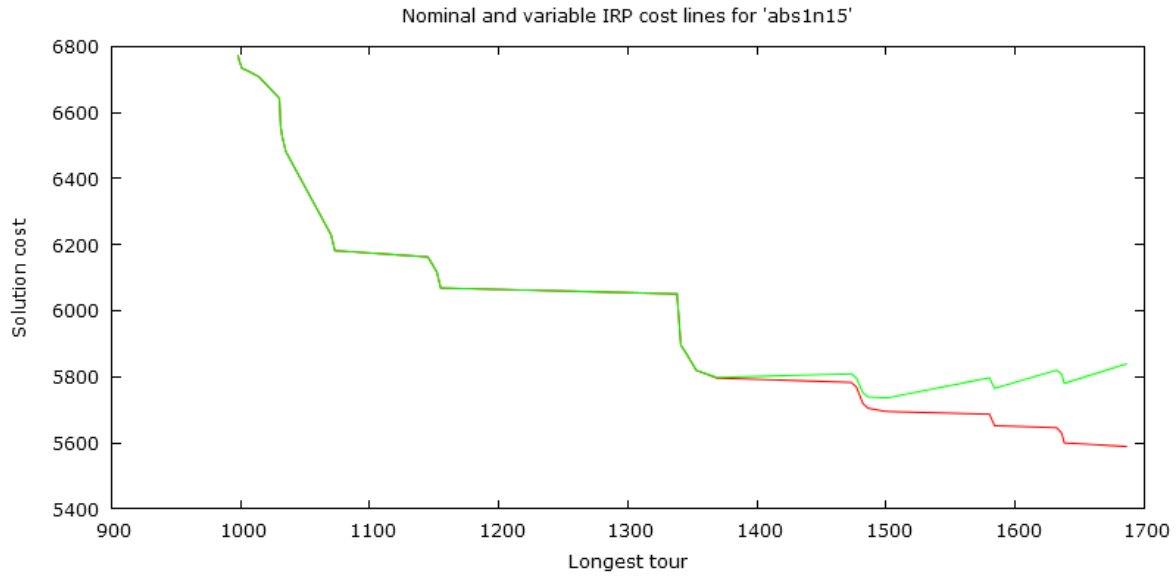

Figure 3.1: Naive approach for the instance 'abs1n15'

3.2.2 Robust approach

A second approach towards finding an optimal solution for the variable IRP is the robust approach. In this approach we will try to optimize the model against some worst case instances that result from uncertainty. Based on our review of the literature on robust optimization (see section 2.3) we chose to use the approach proposed by Bertsimas and Sim. If we take the values of Γ_0 integer in the interval $[0, |J_0|]$, Γ_0 get a specific meaning, namely the number of roads for which we include the worst cost(time) in our model.

A first step in this approach is to rewrite the model. Remind that $\hat{c}_{ij}$ is the maximum

deviation of the nominal value:

$$min \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{i \in V} \sum_{j \in V} \sum_{t \in T} c_{ij} x_{ij}^t + \sum_{t \in T} PenaltyP_t + \max_{S_0 | S_0 \subseteq J_0, |S_0| \leq \Gamma_0} \left\{ \sum_{(i,j,t) \in S_0} \hat{c}_{ij} |x_{ij}^t| \right\} \tag{1}$$

subject to $(2) - (15)$

$$\sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}^t + \max_{S_0 | S_0 \subseteq J_0, |S_0| \leq \Gamma_0} \left\{ \sum_{(i,j,t) \in S_0} \hat{c}_{ij} |x_{ij}^t| \right\} \leq TimeLimit + MP_t \ \forall t \in T \tag{16}$$

$$P_t \in \{0, 1\} \ \forall t \in T \tag{17}$$

Now we can transform the model to a linear optimization model as in section 2.3. Starting by writing the domain of the maximalisation problem as constraints:

$$min \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{i \in V} \sum_{j \in V} \sum_{t \in T} c_{ij} x_{ij}^t + \sum_{t \in T} PenaltyP_t + \max \left\{ \sum_{(i,j,t) \in S_0} \hat{c}_{ij} |x_{ij}^t| \right\} \tag{1}$$

subject to $(2) - (15)$

$$\sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}^t + \max \left\{ \sum_{(i,j,t) \in S_0} \hat{c}_{ij} |x_{ij}^t| \right\} \leq TimeLimit + MP_t \ \forall t \in T \tag{16}$$

$$P_t \in \{0, 1\} \ \forall t \in T \tag{17}$$

$$|S_0| \leq \Gamma_0 \tag{18}$$

$$S_0 \subseteq J_0 \tag{19}$$

Next we can explicit the selection out of set S_0. As shown in section 2.3 selection of the subset is equal to solving the subproblem:

$$\begin{aligned} max & \sum_{(i,j,t) \in J_0} \hat{c}_{ij} |x_{ij}^t| z_{0ij}^t \\ subject\ to & \sum_{(i,j,t) \in J_0} z_{0ij}^t \leq \Gamma_0 \\ & 0 \leq z_{0ij}^t \leq 1 \ \forall (i,j,t) \in J_0 \end{aligned}$$

Note that this subproblem is the same for the constraints as for the objective function, since

the same coefficients are used. Substituting this into the original program we have:

$$min \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{i \in V} \sum_{j \in V} \sum_{t \in T} c_{ij} x_{ij}^t + \sum_{t \in T} PenaltyP_t + \max \left\{ \sum_{(i,j,t) \in J_0} \hat{c}_{ij} |x_{ij}^t| z_{0ij}^t \right\} \tag{1}$$

subject to $(2) - (15)$

$$\sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}^t + \max \left\{ \sum_{(i,j,t) \in J_0} \hat{c}_{ij} |x_{ij}^t| z_{0ij}^t \right\} \leq TimeLimit + MP_t \ \forall t \in T \tag{16}$$

$$P_t \in \{0,1\} \ \forall t \in T \tag{17}$$

$$\sum_{(i,j,t) \in J_0} z_{0ij}^t \leq \Gamma_0 \tag{18}$$

$$0 \leq z_{0ij}^t \leq 1 \ \forall (i,j,t) \in J_0 \tag{19}$$

Now we will introduce the dual variables for the subproblem to linearize the problem. The constraints that are dualized are constraints (18) and constraints (19). Dual variables are respectively z_0 and p_{0ij}^t:

$$min \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{i \in V} \sum_{j \in V} \sum_{t \in T} c_{ij} x_{ij}^t + \sum_{t \in T} PenaltyP_t + \sum_{(i,j,t) \in J_0} p_{0ij}^t + \Gamma_0 z_0 \tag{1}$$

subject to $(2) - (15)$

$$\sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}^t + \sum_{(i,j,t) \in J_0} p_{0ij}^t + \Gamma_0 z_0 \leq TimeLimit + MP_t \ \forall t \in T \tag{16}$$

$$P_t \in \{0,1\} \ \forall t \in T \tag{17}$$

$$z_0 + p_{0ij}^t \geq \hat{c}_{ij} |x_{ij}^t| \ \forall (i,j,t) \in J_0 \tag{18}$$

$$z_0 \geq 0 \tag{19}$$

$$p_{0ij}^t \geq 0 \tag{20}$$

A final simplification can be made by replacing $|x_{ij}^t|$ by x_{ij}^t. This is justified since the variables

x_{ij}^t are binary:

$$min \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{i \in V} \sum_{j \in V} \sum_{t \in T} c_{ij} x_{ij}^t + \sum_{t \in T} PenaltyP_t + \sum_{(i,j,t) \in J_0} p_{0ij}^t + \Gamma_0 z_0 \quad (1)$$

subject to $(2) - (15)$

$$\sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}^t + \sum_{(i,j,t) \in J_0} p_{0ij}^t + \Gamma_0 z_0 \leq TimeLimit + MP_t \ \forall t \in T \quad (16)$$

$$P_t \in \{0,1\} \ \forall t \in T \quad (17)$$

$$z_0 + p_{0ij}^t \geq \hat{c}_{ij} x_{ij}^t \ \forall (i,j,t) \in J_0 \quad (18)$$

$$z_0 \geq 0 \quad (19)$$

$$p_{0ij}^t \geq 0 \quad (20)$$

Now we obtain a MIP. We can find different solutions by varying Γ_0. The closer Γ_0 is to zero, the less we are protecting against uncertainty. The coser Γ_0 is to $|J_0|$, the more we are protecting against uncertainty and the more conservatif our solution will be.

Our strategy to find the most optimal solution for the IRPVTT is analogue to the one applied in the naive approach. We will first identify interesting solutions by increasing Γ_0 from 0 to $|J_0|$ and afterwards apply Monte Carlo simulation on the found solutions.

In figure 3.2 the result of this strategy is depicted. The red and green lines represent respectively the nominal cost of the different solutions and their cost after Monte Carlo-simulation. The longest tour of the optimal nominal solution is 1686 and we assumed a penalty of 500,0 for violating $TimeLimit$. We can see that the cost for solutions of the nominal problem rises if we enfore shorter tours(line in red). The cost of the variable IRP after Monte Carlo simulation(line in green) is different from the cost line of the nominal problem. We can clearly see a higher cost for solutions with longer tours. This higher cost is due to the higher probability of violating the $TimeLimit$. For solution with shorter tours the variable IRP cost and nominal IRP cost as almost the same, since there's very little chance of having to pay the penalty. The optimal solution of the variable IRP has a maximum tour length of 1482.

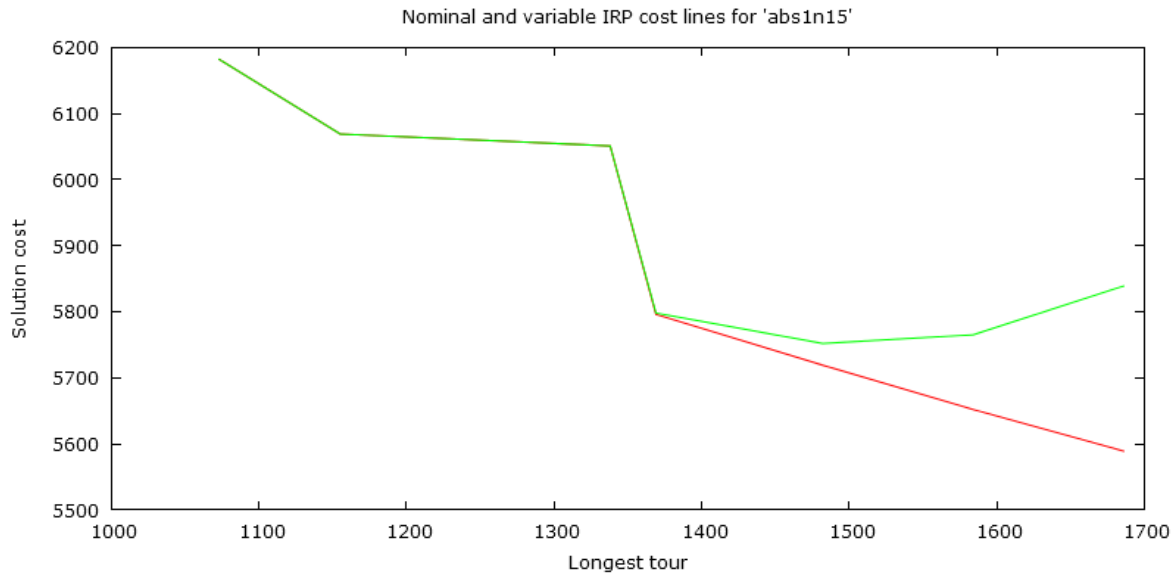

Figure 3.2: Robust approach for the instance 'abs1n15'

3.2.3 Two-phase approach

A third approach we will develop is based on two stages. In the first stage we will solve the IRPVTT based on a set of scenarios S. All these scenarios are represented by a portion of their cost, relative to the number of scenarios, in the objective function and for each scenario we have constraints to model their potential violation of the $TimeLimit$. For every scenario of S we introduce a new variable P_t^p, which is defined as:

$$P_t^p = \begin{cases} 1 & \text{if for scenario p the tour in period t is bigger than the limit } TimeLimit \\ 0 & \text{otherwise} \end{cases}$$

The first stage can be mathematically formulated in the following way:

$$min \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{p \in S} \sum_{i \in V} \sum_{j \in V} \sum_{t \in T} \frac{c_{ij}^p}{|S|} x_{ij}^t + \sum_{p \in S} \sum_{t \in T} \frac{Penalty}{|S|} P_t^p \tag{1}$$

$$subject\ to\ (2) - (15)$$

$$\sum_{i \in V} \sum_{j \in V} c_{ij}^p x_{ij}^t \leq TimeLimit + M P_t^p \ \forall t \in T,\ \forall p \in S \tag{16}$$

$$P_t^p \in \{0, 1\} \ \forall t \in T,\ \forall p \in S \tag{17}$$

Note that the scenarios are only different in transportation costs (and travel times) since these are the only data subject to variation. In the second stage we will look for scenarios,

by varying c_{ij} within the bounds of the uncertainty, having a cost that is very different from the cost associated to the solution of the first stage. Since we only assume uncertain travel times, we can limit the difference between the first stage solution and the new scenarios to the difference in their transportation costs. We will then include these scenarios in the problem of the first stage. Now we can solve the first stage again, probably obtaining a different result after which we can apply the second stage again. Through multiple of such iterations we can find an optimal solution to the IRPVTT.

However, there are some complications in this approach. First the uncertainty defined on the parameters of the model is continuous. To use the approach described above we will need to discretize the uncertainty on some levels. Next we will have to make sure that scenarios are only added once. Finally since we don't know if the scenario with the greatest difference has a greater cost or a lower cost, we will have to solve two programs and take the maximum of both.

Having this in mind we can formulate the following programs in which c_0 to c_n are the different levels of the discretized uncertain parameter $\tilde{c}_{ij}$ and c stands for the vector of all uncertain c_{ij}:

$$max \sum_{i\in V}\sum_{j\in V}\sum_{t\in T} c_{ij}x_{ij}^t + \sum_{t\in T} PenaltyP^t - \left[\sum_{p\in S}\sum_{i\in V}\sum_{j\in V}\sum_{t\in T}\frac{c_{ij}^p}{|S|}x_{ij}^t + \sum_{p\in S}\sum_{t\in T}\frac{Penalty}{|S|}P_t^p\right] \quad (1)$$

$$TimeLimit \leq \sum_{i\in V}\sum_{j\in V} c_{ij}x_{ij}^t + M(1-P_t)\ \forall t \in T \quad (2)$$

$$c \neq c^p\ \forall p \in S \quad (3)$$

$$c_{ij} = \{c_0..c_n\}\ \forall i \in V,\ \forall j \in V \quad (4)$$

$$P_t \in \{0,1\}\ \forall t \in T \quad (5)$$

$$max -\sum_{i\in V}\sum_{j\in V}\sum_{t\in T} c_{ij}x_{ij}^t - \sum_{t\in T} PenaltyP^t + \left[\sum_{p\in S}\sum_{i\in V}\sum_{j\in V}\sum_{t\in T}\frac{c_{ij}^p}{|S|}x_{ij}^t + \sum_{p\in S}\sum_{t\in T}\frac{Penalty}{|S|}P_t^p\right] \quad (1)$$

$$TimeLimit \leq \sum_{i\in V}\sum_{j\in V} c_{ij}x_{ij}^t + M(1-P_t)\ \forall t \in T \quad (2)$$

$$c \neq c^p\ \forall p \in S \quad (3)$$

$$c_{ij} = \{c_0..c_n\}\ \forall i \in V,\ \forall j \in V \quad (4)$$

$$P_t \in \{0,1\}\ \forall t \in T \quad (5)$$

Note that in these programs the part between brackets is constant and equals the average transportation costs and penalty associated with the solution found in the first stage. The

only variables in these programs are c_{ij} and P_t When solving these two programs, we can take the maximum of both. The vector c associated with this difference can then be added as scenario to the first stage. Remark that the constraint $c \neq c^p$ is not a typical MIP constraint. We will discuss this elaborately in section 4.4.1.

In figure 3.3 the results of this approach are depicted. We can see that initially the optimal solution has a maximum tour length of 1141 (corresponding to the optimal solution of the nominal IRP), but changes through the various iterations and finally an optimal solution with a maximum tour length of 907 is found. Contrary to the first two methods no Monte Carlo simulation is needed to evaluate the solutions.

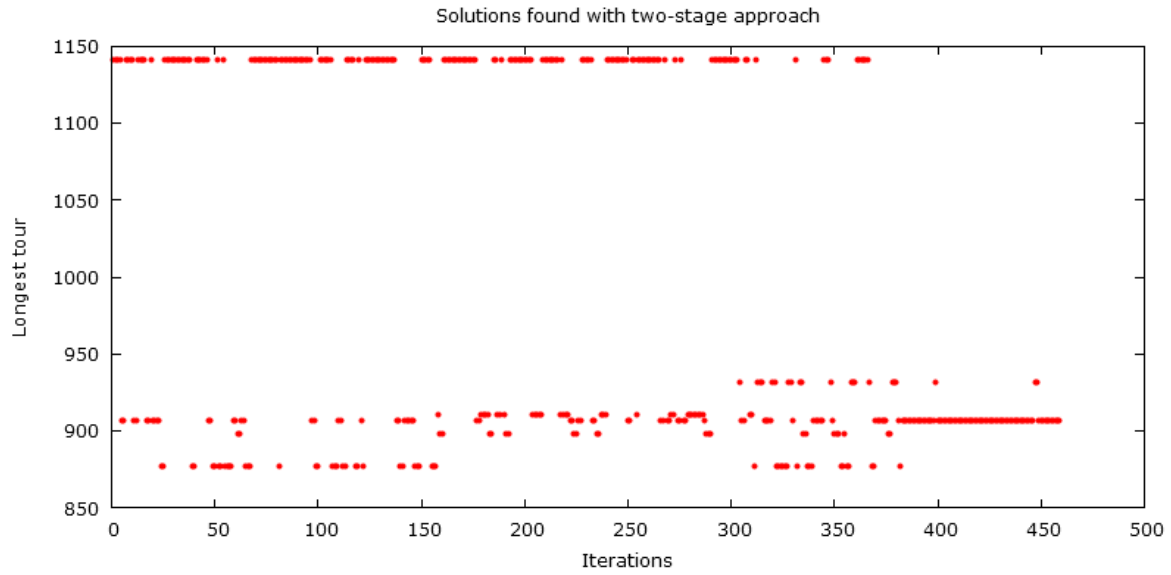

Figure 3.3: Two-stage approach for the instance 'abs1n5'

3.2.4 Scenario Approach

The scenario approach is just like the two phase approach based on a certain set of scenarios S. In contrary to the two-phase approach where worst case scenarios are first added to the set S, the scenario approach will solve the problem for a set of random scenarios.

The MIP corresponding to this approach is given below:

$$min \sum_{i \in V} \sum_{t \in T} h_i I_i^t + \sum_{p \in S} \sum_{i \in V} \sum_{j \in V} \sum_{t \in T} \frac{c_{ij}^p}{|S|} x_{ij}^t + \sum_{p \in S} \sum_{t \in T} \frac{Penalty}{|S|} P_t^p \quad (1)$$

$$subject\ to\ (2) - (15)$$

$$\sum_{i \in V} \sum_{j \in V} c_{ij}^p x_{ij}^t \leq TimeLimit + M P_t^p \ \forall t \in T,\ \forall p \in S \quad (16)$$

$$P_t^p \in \{0, 1\}\ \forall t \in T \quad (17)$$

An example of the scenario approach for the instance 'abs1n5' is given in figure 3.4. In this example we vary the number of scenarios between 1 and 500 and fix the penalty on 500. We can see that in the beginning there's some variation in the optimal solution for the variable IRP but finally it converges to 907.

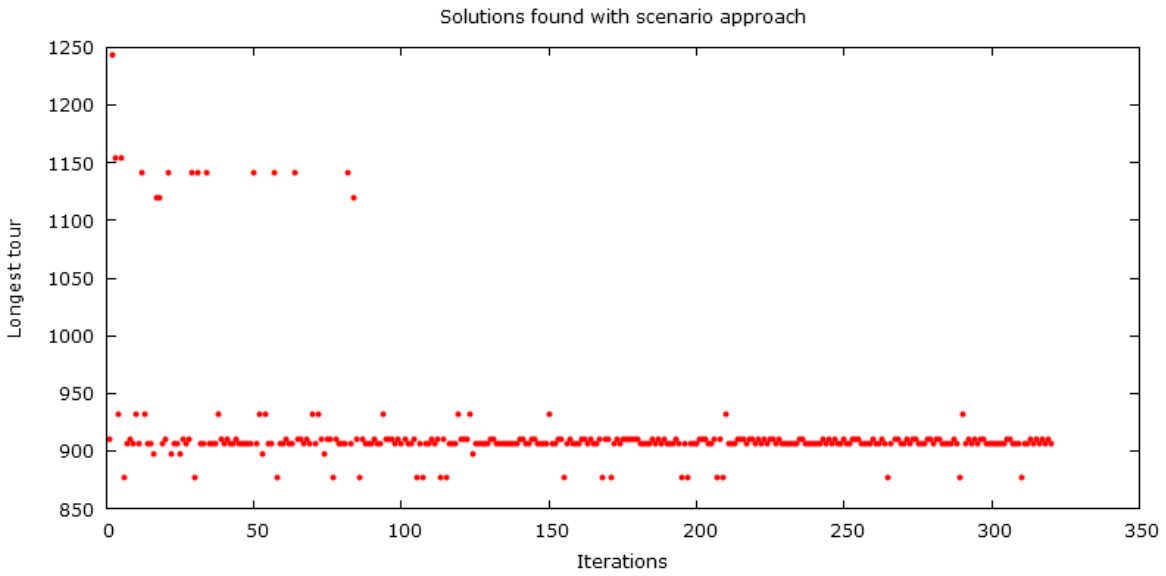

Figure 3.4: Scenario approach for the instance 'abs1n5'

3.2.5 Conclusion

In this section we presented briefly the four proposed approaches. The first two approaches try to identify solutions over the whole range of variability and apply Monte Carlo simulation afterwards to compare the solutions. The last two approaches solve the problem for a set of scenarios. The comparison between different solutions happens thus directly.

Chapter 4

Detailed development of the approaches

In this section we will evaluate the approaches presented in the previous section and evaluate their solution both in terms of accuracy and solution time.

4.1 General Improvements

In what follows we will discuss improvements that aim at increasing the accuracy and decreasing the execution time of the methods. In all methods iterations are used in one way or another: either for front construction, to solve the first stage problem or to solve the scenario approach problem. Due to the subtour elimination constraints the MIP formulation is of exponential size. In our implementation subtour elimination constraints are added dynamically when needed. Since we solve very similar problems throug various iterations we can keep the subtour elimination constraints of the previous iterations. This will avoid redoing all the work done in previous iterations and will speed up all methods.

4.2 Naive Approach

Before analysing this method in detail we can already sum up some general drawbacks of the approach. First there's the inefficiency of the method. The problem has to be solved several times to identify different interesting solutions and there is also the need to do the Monte Carlo simulation afterwards. Second it is not given that the optimal solution to the IRPVTT is for sure one of the interesting solutions identified in the first step.

4.2.1 Limiting Front Construction

We will first focus on the inefficiency of the method. When we look at the solution time for solving the problem with different values of $TimeLimit$(we denote each new $TimeLimit$ as a new iteration in the procedure) we see in figure 4.1 that the solution time increases exponentially when lowering $TimeLimit$(i.e. going to the next iteration). For the instance with 20 clients we see that the solution time reaches after a few iterations the execution time limit of 30 minutes (1800 seconds) we imposed on the calculations. In this case the MIP Solver will give a solution but without prove of optimality.

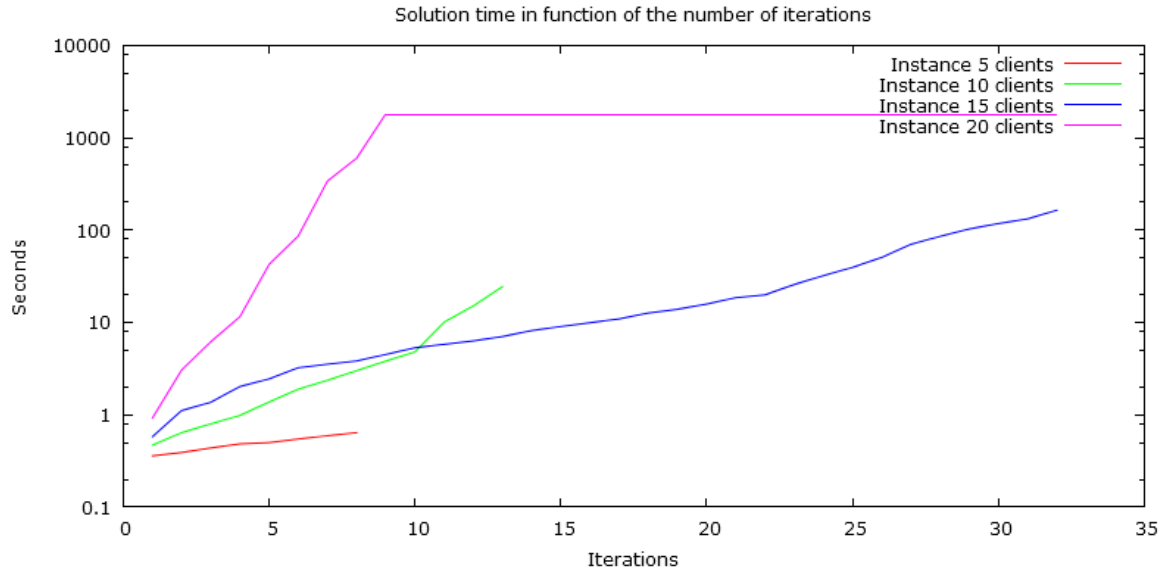

Figure 4.1: Naive approach: Solution time for different $TimeLimit$

An example of a case where the time limit is reached is shown in figure 4.2. The part left of the the vertical blue line shows the solutions where the execution time limit is reached. We can clearly observe a very instable behaviour. The function also doesn't have its monotone decreasing form any more. We can approximate the front of optimal solutions by connecting the points with the lowest costs to each other, as seen by the green line. We can expect that the behaviour will even worsen for bigger instances and that the blue line will shift to the right.

The question now raises: Do we always need the whole front to find the optimal solution to the variable IRP? The answer is no. If the deviation is known in advance the front construction can stop when $TimeLimit$ >Longest tour in solution * (1 + maximum deviation). The reason is that when this condition is statisfied the probability of paying a penalty is zero. Knowing that enforcing smaller tours will only increase nominal costs, we can safely stop when this

bound is reached. In figure 4.2 this part bound is shown with a purple line. In this example the *TimeLimit* is 1760 and the maximum deviation is 50%. The longest tour will thus be greater than 1173,33. Every solution left to the purple line is thus eliminated. In this example this only eliminates 1 solution, but we can go further.

If we apply Monte Carlo simulation after every iteration of the front construction we can stop the front construction when we find a solution to the nominal IRP whose costs are higher than the costs of the current best solution to the variable IRP. In figure 4.2 we see that after tree iterations a solution with a longest tour of 1660 and a cost of 7320,9 is found for the variable IRP. In the fourth iteration we find a solution with a tour of 1639 and nominal costs of 7179,52. This is not enough to stop the front construction, thus we continue. In the fifth iteration we find a solution with a longest tour of 1622 and nominal costs of 7390,01. Since the nominal costs are higher than costs of the variable IRP, we can now stop the procedure. By doing the Monte Carlo simulation right after every iteration of the front construction we avoid to spend time on the hard problems with (enforced) lower tours.

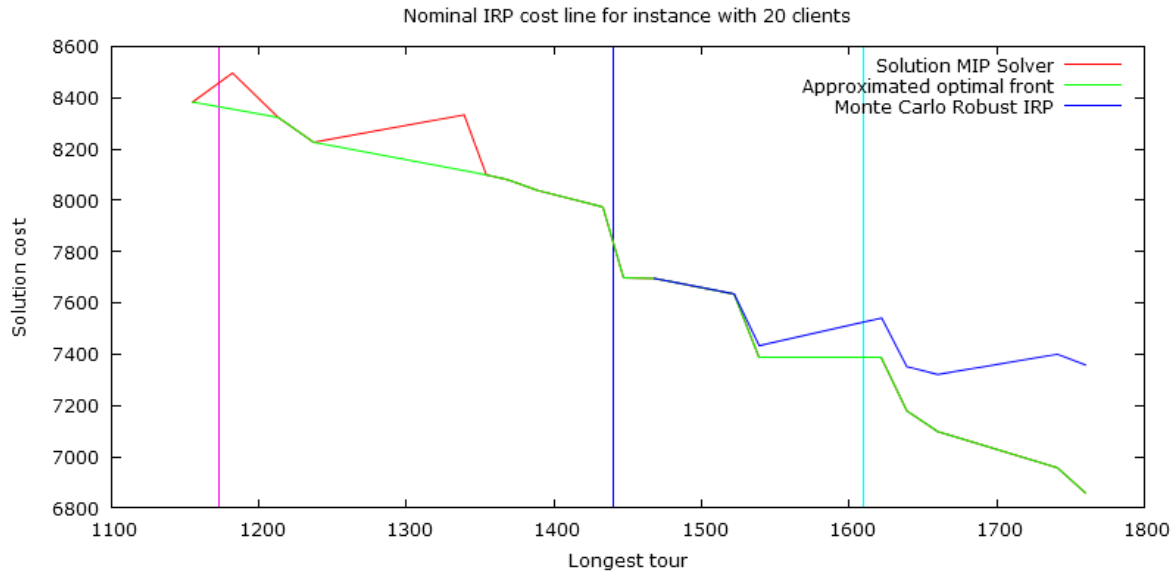

Figure 4.2: Naive approach: Instable behaviour for big instances

Evaluation of execution time

To make a detailed analysis of the quality of the solution in terms of execution time is difficult, since we stop the procedure after the interesting part which depends on the height of the penalty and the maximum deviation. However, we can state some general comments about the procedure. For instances with a small number of clients(5-15 clients) the procedure can find the whole front of optimal solutions for different *TimeLimit* in reasonable time(0,6-90

s). For bigger instances (20-50 clients) some iterations will reach the solution time limit of 1800 s and the front of optimal of optimal solutions will be approximated. After the solutions are identified Monte Carlo-simulation can be conducted. In table 4.1 the execution time of both front construction for the whole front and Monte Carlo-simulation of one solution with 1.000.000 samples are shown.

Number of clients	5	10	15	20	25	30
Front Construction[s]	0,6	27,58	89,52	>18.000(5h)	-	-
Monte Carlo Simulation[s]	7,56	24,52	38,86	42,32	66,96	89,02

Table 4.1: Execution time of Monte Carlo simulation and front construction in function of the number of clients for the naive approach

Whereas the execution time for front construction is clearly increasing very fast due to the increasingly hard problems, the execution time for Monte Carlo-Simulation is increasing rather slowly. The relatively small time for Monte Carlo simulations gives the approach a great advantage. Once the front is fully constructed, we can evaluate solutions for every value of $TimeLimit$ and penalty since the front does not depend on these parameters.

4.2.2 Evaluation of accuracy

In terms of accuracy we can say that due to the Monte Carlo simulation, it is highly likely that the best solution will be found among the ones identified. However, the optimal solution for the variable IRP is not necessarily also one of the solutions for the nominal IRP. Due to this fact the procedure may lead to a wrong result. We belief however that when the optimal solution is found in a region where the density of solutions found is rather big, the probability that there's such a 'hidden' optimal solution is small. When such a solution would exist the difference in cost can be at most the absolute value between the cost of the optimal solution found and the cost of the optimal solution of the nominal problem.

4.2.3 Conclusion

As conclusion we can state that this approach is suited for small instances and will give, with a high probability, accurate results. Since the front construction procedure shows instable behaviour for larger instances, it is little suited to solve larger instances. Note that for larger instances some cases can still be solved reliably when the optimal solution is found the reliable region. In the next section we will look at a method which may shorten the time of the front construction. The sections after that will be dedicated to methods who can solve the problem for larger instances.

4.3 Robust Approach

Since the robust approach uses in general the same strategy as the naive approach, namely identifying interesting solutions and then applying Monte Carlo-simulation to them, we can assume that it will have some similar drawbacks. First there is the drawback that the problem has to be solved for different values of the parameter Γ_0 during the front construction and second there's the need for Monte Carlo simulation afterwards. The front construction will play the most important role in the execution time.

4.3.1 Evaluation of the front construction

Basically, there are two main effects that will determine the execution time of the front construction. The first effect is that compared to the naive approach the robust approach will in general require less executions of the program. This is a result of finding a solution for a certain value of Γ_0 in which z_0 equals 0. At this point raising Γ_0 will have no influence, since the objective function no longer depends on Γ_0, it is multiplied by 0, and the procedure can thus be stopped. This behaviour is shown in the figure 4.3.

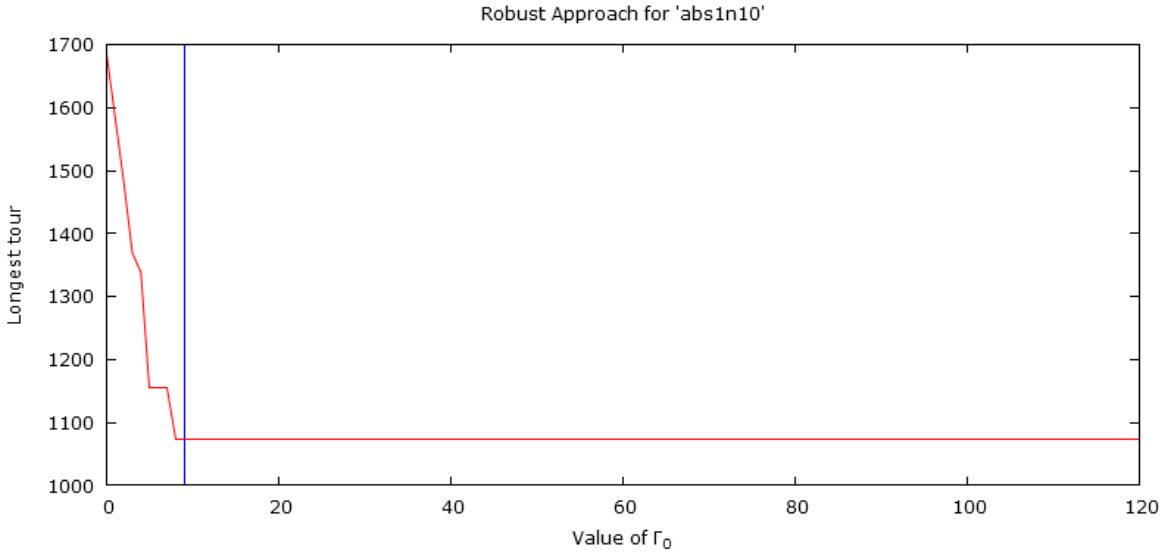

Figure 4.3: Robust approach on instance 'abs1n10'

This happens because the robust approach always takes the Γ_0 worst parameters into account. Whenever Γ_0 equals the number of unique arcs that appear in the solution(and the influence of *Penalty* is considerable), the program will put z_0 on zero. The stopping criterion is thus $\Gamma_0 >$ the number of unique arcs appearing in the solution. This stopping criterion is related

to the one of the naive approach, $TimeLimit$ >Longest tour in solution * (1 + maximum deviation). If Γ_0 is greater than the number of unique arcs in the solution, this means that all arcs will have maximum coefficients, equal to their nominal coefficient * (1 + maximum deviation). Under a considerable $Penalty$ every tour will be shorter than $TimeLimit$ and thus will the condition $TimeLimit$ >Longest tour in solution * (1 + maximum deviation) be satisfied.

The second effect that is playing is the increase in variables and constraints. Due to these added variables and constraints, the robust program takes longer to identify interesting solutions than the naive approach. For an instance with 15 clients the execution time of the first 10 iterations is given in table 4.2. Compared to the naive approach we can clearly see, in terms of time, the cost of the added variables and constraints.

Iteration	1	2	3	4	5	6	7	8	9	10
Naive approach[s]	0,31	0,25	1,90	0,47	0,35	0,34	0,32	0,45	0,70	0,52
Robust approach[s]	0,37	0,58	1,32	1,53	5,12	5,27	5,21	10,61	7,91	20,72

Table 4.2: Execution time per iterations for the front construction

4.3.2 Evaluation of execution time

So on one hand we have a reduction of iterations in the front construction and on the other hand we have a longer duration for each iteration. When we look at the final results in table 4.3, we see that for the small instances the front construction of the robust approach goes faster than the front construction of the naive approach. However for large instances we can expect that it will take much more time. We can already see this for the instance with 20 clients.

Another disadvantage of the robust approach is also the fact that the front construction depends on the values of $TimeLimit$, $Penalty$ and maximum deviation. We lose thus the independency between the front construction and the Monte Carlo-Simulation we had in the naive approach. For every change in these parameters, the front will have to be constructed

Number of clients	5	10	15	20	25	30
Front Construction[s]	0,5	17,89	59,53	>30000(9h)	-	-
Monte Carlo Simulation[s]	7,56	24,52	38,86	42,32	66,96	89,02

Table 4.3: Execution time of Monte Carlo simulation and front construction in function of the number of clients for the robust approach

again.

4.3.3 Evaluation of accuracy

Until now we have only examined the robust approach in terms of time. Regarding the accuracy of the method we can expect that compared to the naive approach the robust approach will be less accurate. The front construction has less iterations and this results into a rougher approximation of the Pareto front and the variable IRP cost function. We can see this effect in figure 4.4. Furthermore the argument that an optimal solution might not be idenified in the first step still holds.

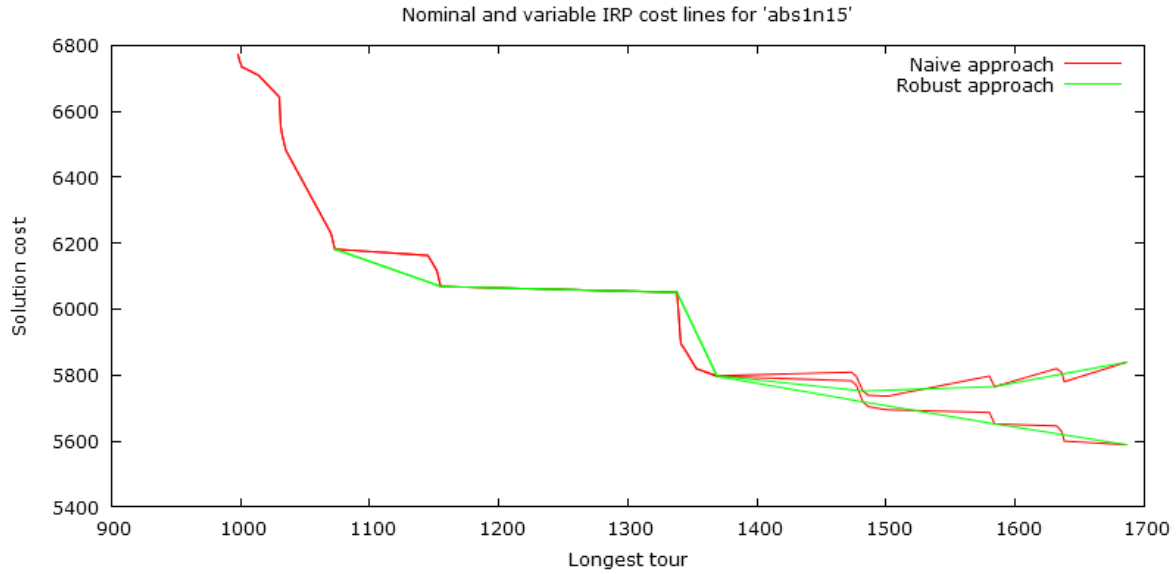

Figure 4.4: Robust approach approximation

Finally we mention also one anomaly that may occur when using this approach. This anomaly is due to the original formulation of the problem. When an optimal solution is found in which the tour of periode t consists of visiting one node j, then the program will put one of the variables x_{0j}^t on 2. All other transport variables $x_{ij}^t, i \neq 0$ can at most have a value of 1. During the robust approach it may occur that optimizing for a value of Γ_0 results in protecting against x_{0j}^t which is used in a 2-tour. Increasing Γ_0 will result in a lower cost, even though this means a higher level of conservatism. In this case increasing Γ_0 will result in another solution in which the 2-tour doesn't appear. The anomaly appears because for the first value of Γ_0, the costs is added twice, since x_{0j}^t equals 2. In the next iteration a higher cost is added, but only one time.

A possible solution to avoid this anomaly is breaking the formulation of the IRP, so x_{ij}^t is

always binary. However, this has a considerable impact on the solution time of the problem. Therefore we will use the current robust formulation, paying extra attention to possible occurences of this anomaly.

4.3.4 Conclusion

As a conclusion we can state that we are confronted with similar problems as in the naive approach. The different way of constructing the front does not solve the exponential increase in time, it worsens it. The method is thus not at all suited for larger instances. Furthermore, the approximation of the front results in a decrease of accuracy which makes the method less attractive than the naive approach. In the next sections we will explore methods that approach the problem totally different and that will prove to be more suited for larger instances.

4.4 Two-phase Approach

The performance of the two-phase approach will depend on both the first and second stage of the solution. To further develop this method we will first focus on the second stage, then on the first stage and finally on the interaction between both.

4.4.1 Approximating the second stage

As said in section 3.2.3, the second stage has a particular constraint: $c \neq c^p$. This constraint is needed to ensure that the same scenario is not added more than one time. However, this constraint can not be translated into a constraint of the form $>$, $<$or $=$. Therefore solving a problem having such a constraint is not that straightforward. A possible solution method is based on branch- and bound. First we remove the constraint $c \neq c^p$ and solve the relaxation of the first second-stage problem(case of the second second-stage problem is analogue). If $c \neq c^p$, the problem is solved. If not, we take a variable c_{ij} and solve the relaxation once with the constraint $c_{ij} < c^p_{ij}$, once with $c_{ij} > c^p_{ij}$ and once with $c_{ij} = c^p_{ij}$. Now we can solve these problems and continu the same strategy. We can also calculate the bounds for the branches $c_{ij} < c^p_{ij}$ and $c_{ij} > c^p_{ij}$, given that the constraint $c \neq c^p$ is not violated, and apply pruning based on these bounds in the branches. Note that we can't prune based on a bound of the branch $c_{ij} = c^p_{ij}$, since this branch constains nodes that violate $c \neq c^p$. This strategy is depicted in figure 4.5.

Remark that this strategy can be very time consuming. We know that the number of c_{ij} is exponential in function of the number of clients and for each c_{ij} we have different levels from c_0 to c_n. In total there are $\frac{(N+1)^2-(N+1)}{2}$ variables c_{ij} and every variable can take all values

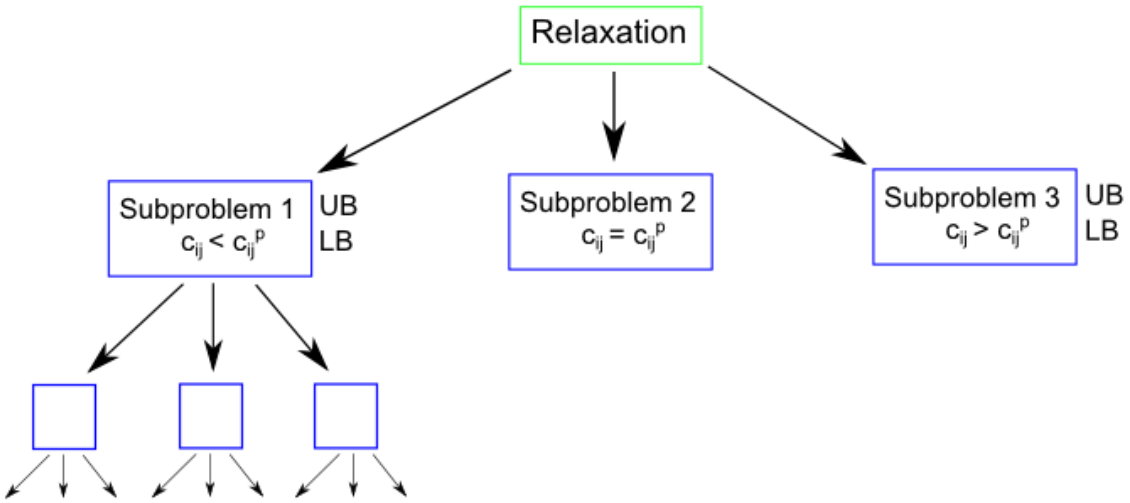

Figure 4.5: Rigourous strategy to find worst scenario

in the range c_0 to c_n. The total number of scenarios is thus $M^{\frac{(N+1)^2-(N+1)}{2}}$ for a case with N clients and M levels. For the case of 5 clients and 3 levels, this already results in 14.348.907 possible scenarios.

Now the question emerges whether it is really necessary to find the most optimal vector c. The vector c is used in the first stage to be part of the set of scenarios on which the first-stage problem is optimized. One scenario is only adding little, since its contribution to the objective function is relative to the number of scenarios. It might not really be necessary to find the exact worst case scenario, but a similar scenario would be a good approximation. We are also not excluding the exact worst case scenario, so it is still eligible in further iterations. The huge advantage of taking a 'bad case' scenario, without the proof that it is worst case, is the enormous simplification of the problem.

As we saw in figure 4.5 finding the worst case scenario is equivalent to going through a tree. Now instead of going through the whole tree we can solve the relaxation. If we see that this results in a vector c that is already in the set of scenarios, let say it corresponds to c^p, we can add a dynamically a constraint on a conflicting c_{ij} forcing it to be bigger or smaller than c_{ij}^p, keeping in mind the limits c_0 to c_n of c_{ij}. Now we can solve the problem again. If we find a new solution different from any scenario in the current set, the problem is solved. If not we can, find a new conflicting variable c_{ij} and repeat the procedure above. Instead of looking at the whole tree, we will just look at one path of the tree.

This will procedure will provide us with a solution to the second-stage problem. However, because of bad luck the solution might be very far from the worst case. Since the second-stage

problem is a rather easy problem. We can repeat the procedure several times and take the worst scenario found. Now we can write down the whole procedure down rigourously:

Algorithm 2 Finding 'bad case' scenarios

loop N times
  Start with the nominal problem s ▷ Step 1
  Solve the problem s ▷ Step 2
  if solution c is part of the set S **then**
    Find a conflicting variable c_{ij}
    Add a constraint so that $c_{ij} > c_{ij}^p$ or $c_{ij} < c_{ij}^p$ to s
    Return so step 2
  else Start the next iteration
  end if
end loop
Take the worst scenario of the scenarios found and add it to S

Now we have made the procedure for finding the vector c faster. As a prelimenary result we can test the two-stage approach in which we run the procedure in the second stage 1000 times. The execution time for the first stage and the second stage is shown in figure 4.6. As we can see the execution time of the second stage is relatively high in the beginning, but stays almost constant, around 4 s, during the whole execution. The execution time of the first stage increases exponentially(at least in the beginning, it seems to converge near the end). The total time of the first stage is 61173,59 s(around 17h), while the total time of the second stage is 3970,77 s(around 1h and 6 minutes). In total the execution time of the first stage determines the overall execution time.

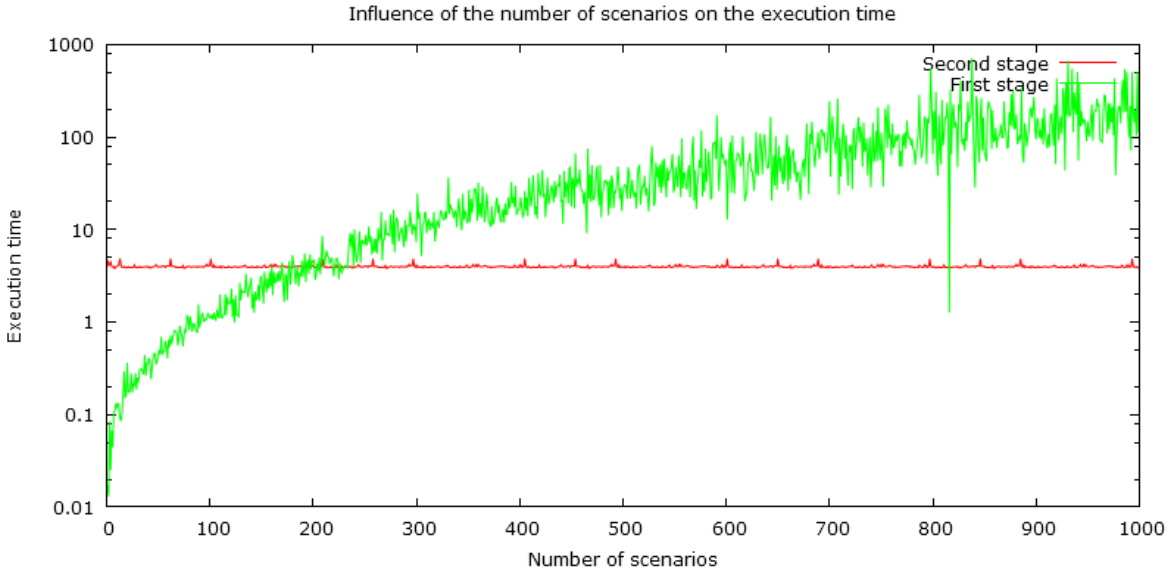

Figure 4.6: Execution time first and second stage

We will further introduce measures to try to cope with the exponential increase in the execution time of the first stage, but a measure we can introduce in the second stage is to choose the scenarios in an interesting way and so try to keep the total set of scenarios as little as possible.

In this regard we propose an interesting measure that is based on the idea that the transportation coefficients c_{ij} of variables x_{ij}^t that are never(i.e. in none of the periods) used in the first stage solution can hold their nominal value in the second stage. Since they are not used in the first stage solution the problem is not sensitive to changes in these coefficients. Because the problem is not sensitive to changes in these coefficients, their values are of little importance and thus they can take any value. We will fix these coefficients on their nominal value, also their average value. By doing so we can reduce the number of scenarios drastically.

For an instance with 5 clients and 3 discretization levels, we calculated that the number of possible scenarios is 14.348.907. The optimal solutions for the instance with 5 clients we identified in the naive approach all use between 5 and 7 different x_{ij}^t. This means that for every solution there are between $3^5(243)$ and $3^7(2187)$ scenarios. We identified 8 possibly interesting solutions. In worst case we have 17.496 solutions which is already a huge contrast with the 14.348.907 scenarios we had before. Remark that in reality the number of scenarios will even be lower since some scenarios will be appear in multiple sets,those will only have to be added once, and not every possibly interesting solution will effectively be interesting, so scenarios only linked to this solution will never be included in the set of scenarios.

In figure 4.7 an example of this approach is given. In red we can see that initially the optimal

solution has a maximum tour length of 1141(corresponding to the optimal solution of the nominal IRP), but changes through the various iterations and finally an optimal solution with a maximum tour length of 907 is found. In green the absolution solution value of the second stage(corresponding to the worst scenario) is depicted. We can see that the general trend is decreasing. We can also clearly distinguish the scenarios linked to the different solutions. The occurences where the value of the second stage differs greatly from the previous values, correspond with the transitions to another solution. Notice that in total we only need 459 scenarios, a lot less than the worst case of 17.496 possible scenarios.

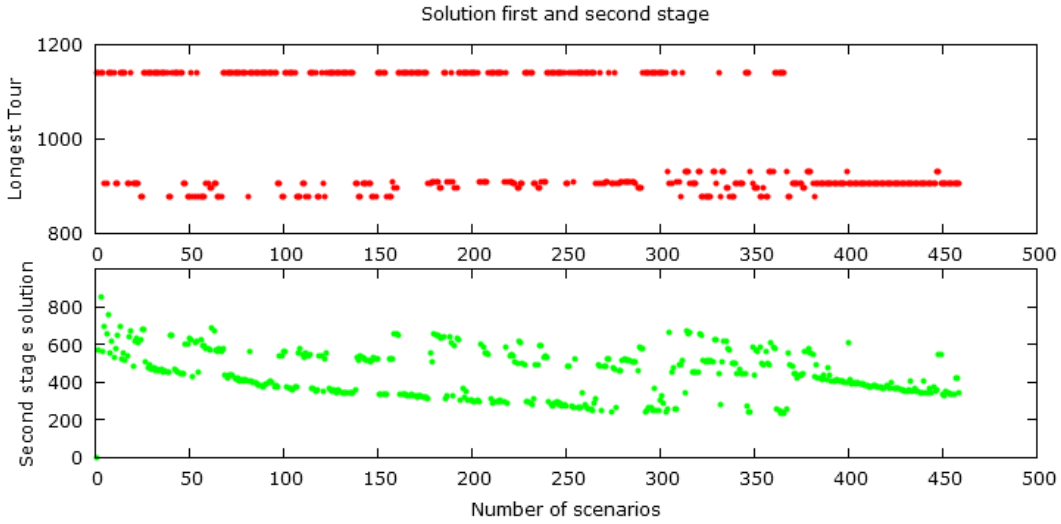

Figure 4.7: Solution first and second stage

4.4.2 Improving the first stage

Now some improvements are made to speed up the second stage and to limit the size of the scenario set, we will concentrate on improvements in the first stage of the approach. The goal of our improvements should be to stop or slow down as much as possible the exponential increase in execution time of the first stage as can be seen in figure 4.6. The main reason why we experience this exponential increase is the increase in variables and constraints. To cope with this increase we could try to fix some variables that are not likely to be interesting.

Fixing on 1 or 2

A very easy improvement that can be taken based on the idea of fixing variables is to fix the transport variables x_{ij} that appear in almost every interesting solution to the value they take,

1 or 2. To identify these variables we can solve the problem for a different sets of scenarios. If we compare all solutions, we can easily identify the x_{ij} appearing in every solution.

A question that emerges in this strategy is how the sets of scenarios should be chosen and how many scenarios should be chosen to identify the transport variables that appear in almost every interesting solution. If the sets are too small, the solutions can be particular solutions for certain particular cases. To much of these solutions will anneal the strategy and no common used transport variable will be found. On the other hand, bigger sets will require more time so that eventually the strategy costs more time than it gains. A similar conflict appears in the number of sets used. A small number will possibly identify too many transport variables as common and will so eliminate other interesting solutions in the future, while a large number of sets increases the time to run the strategy. We are thus twice confronted with a conflict between accuracy and gain in time.

To solve these conflicts we will first remark that in general the two-stage approach solution converges to the optimal solution. As we can see in figure 4.7 in the initial iterations the optimal solution changes but eventually converges towards the optimal solution. If we would thus take a part of the iterations as the sets on which we identify common transport variables, we could resolve some of the issues described above. First no, or not too many, particular solutions will appear since we are converging to the optimal solution. Also all time issues dissapear since we are not adding an extra step to identify common transport variables before the procedure starts, but while it is running. Now one major question remains: In which iterations are we identifying the common variables? The answer to this question will give us both the range of set size and the number of sets.

The most interesting iterations to use are the initial iterations, since in this way the method will have the biggest gain in time. The solutions of these initial iterations will heavily depend on the first worst-case scenarios, so one might suspect that particular solutions will appear here. Therefore it might be better to skip some of the first iterations who mark the warm-up period of the method. To identify the warm-up period we will use the value of the objective function. As can be seen in figure 4.8 the objective function converges relatively quickly. We will define the end of the warm-up period when the relative difference between two successive objective values is less than 1%. As number of sets we will take 100 solutions. So 100 iterations after the warm-up period we will evaluate if there are transport variables that appear in all solutions. If such variables are identified, we will fix them on the value they take. Notice that we can continue to apply this procedure afterwards. Since the solution is converging to the optimal, more of the variables will take their final value. Whenever we notice that for 100 iterations a variable takes the same value we can fix it. In this way we will reduce the number of possible solutions.

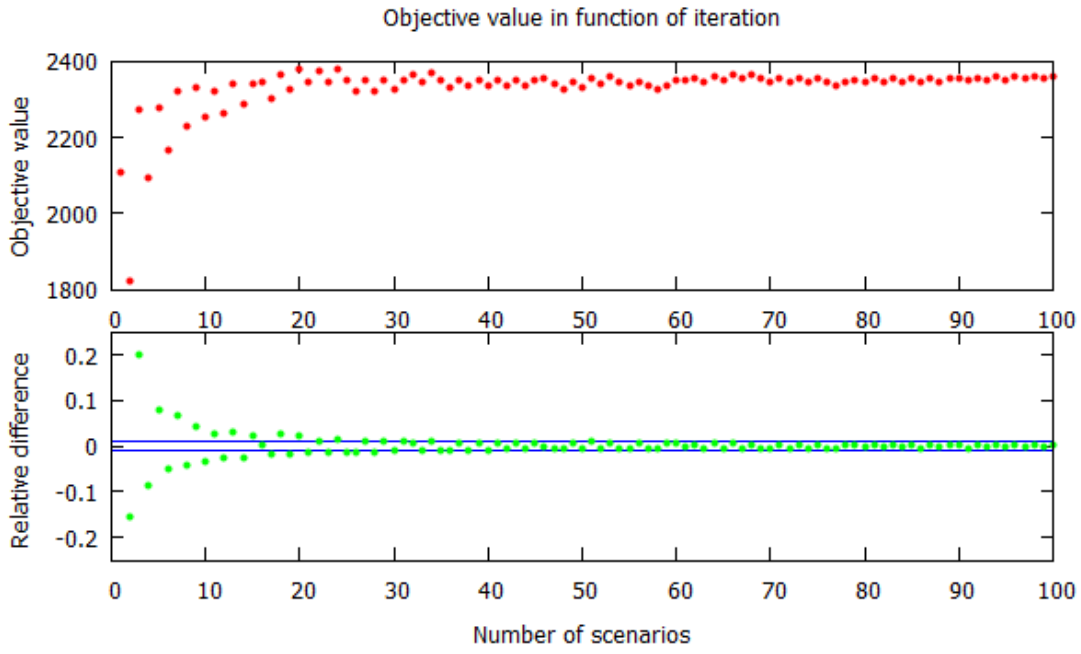

Figure 4.8: Convergence of objective value

The whole approach is depicted in figure 4.9. In the most above graph we see that the solution still converges to a solution where the longest tour is 907. In this example we solve the second stage 100 times. In the graph in the middle we can very clearly see the decreasing tendence of the second stage problem, but due to the smaller loop it is less precise than the one of figure 4.7. In the most below graph we see the effect of our new strategy. We can clearly see that around iteration 109, iteration 168, iteration 481 and iteration 629 the execution time of the first stage drastically decreases. In these iterations variables are fixed. Furthermore we observe the lower time of the second stage. To solve it 100 times it takes around 0,4 s. This is a quite logic result since we saw in 4.6 that it takes around 4,0 s to solve the second stage problem 1000 times. When we add up all the time it took to solve the first stage problems, we get a value of 975,65 s. This is a in huge contrast with the 61173,59 s we obtained earlier. We remark that due to the randomness in the second stage the two solutions have a different number of iterations and are thus not perfectly comparable. However, we feel that the difference both graphically observable as in order of magnitude is clear between the strategies. We remark also that the time to solve the the second stage problems is no longer neglectable: 256,1 s.

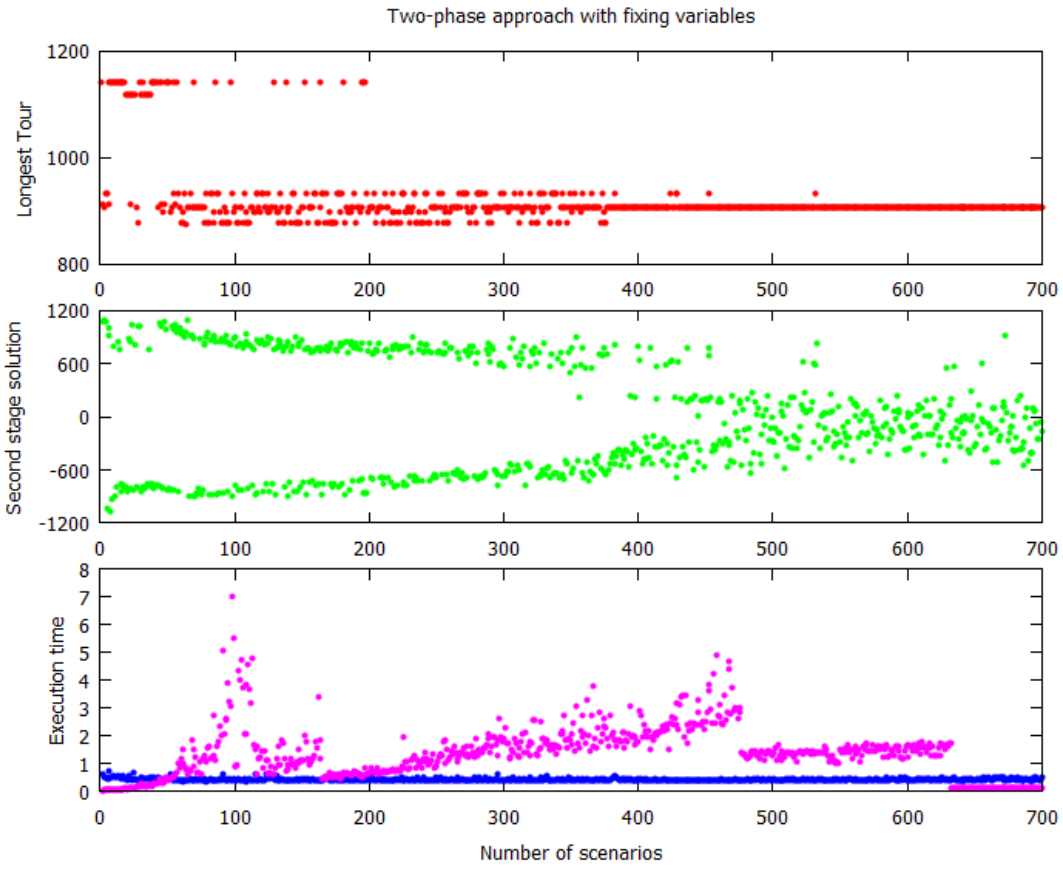

Figure 4.9: Two-phase approach with fixing variables on 1

Fixing on 0

We can now use these results to further improve the method. If we examine the most above graph in 4.9, we observe that from iteration 100 to the end the dominant solution a maximum tour length of 907 has. However, sometimes we see other solutions appear, often only for very few iterations. The reason of this behaviour is due to the fixing of some variables c_{ij} to their nominal value as explained in the beginning of this section. We only allow the variables c_{ij} to change when they appear in solution of the first stage. By keeping some variables c_{ij} to their nominal value, we make solutions using these c_{ij} attractive. When such a solution is found, the second stage will find scenarios that vary the same variables after which the solution becomes less attractive. After a while when the variables are kept to their nominal value again, their attractiveness increases and they will appear again as solution of the first stage. We can see this 'hopping' behaviour very clearly in 4.9.

To avoid this kind of behaviour we will try to eliminate these solution that only appear to be

discarted right after. We will do this by using the opposite strategy of the one described above: fix transport variables to 0. We will not use the same method as for the strategy above. The reason is that the dynamics of the problem are different. In the previous paragraphs we tried to identify common transport variables. We need a lot of iterations before we can identify a common transport variable. However, here we are more interested in how much a certain variable appears, how sensitive is the problem to a certain variable. For example if a variable appears once every 50 iterations, the problem is probably not very sensitive to this variable. With our previous method we would not fix the variable, while in reality we would like to fix the variable and so eliminate an unattractive solution. We will thus need a method a bit more flexible than the previous one.

Instead of evaluating the variables for the last 100 iterations, we will assign a score to each variable. If the variable appears in a solution we will increase the score. If the variable doesn't appear we will decrease the score. If the score drops below a certain threshold, thus meaning that is appearing rarely, the variable will be fixed to 0. Now we have to define the rules to increase and decrease the score.

Choosing the rules to increase and decrease the score symetrically is not a good idea. The reason is that in the beginning the set of scenarios is small and it might take a while before the method finds the optimal solution and even then this optimal solution may only be visited rarely. It is more interesting to increase the score proportially higher than the decrease of the score. In this way variables that are not frequently used will be able to survive a certain period. If the period is too long, they will be fixed to zero. Keeping this in mind we choose to assign an initial score of 1 to each variable and we will use the following rules:

$$score_{ij}^{t} = \begin{cases} \max\{2\ score_{ij}^{t};\ 1\} & \text{if variable } x_{ij}^{t} \text{ is used} \\ 0,95\ score_{ij}^{t} & \text{otherwise} \end{cases}$$

If we notice the score of a variable is under 0,01, we will decide to fix this variable to 0. In figure 4.10 the gain in time by using this strategy is depicted(only the first stage execution time is shown). The vertical lines in darkblue represent moments when variables are fixed to 0. The vertical lines in purple represent moment when variables are fixed to 1 in the strategy that fixes variables to both 0 and 1. The vertical lines in lightblue represent moments when variables are fixed to 1 in the strategy that only fixes variables to 1. We can clearly see that the time required by an iteration is less for the strategy of fixing variables to both 0 and 1. In general we see that fixing variables to 1 doesn't happen that frequently(the purple and lightblue lines). Compared to the fixing to 1, the fxing to 0 are far more frequent. These measures result in the biggest gain in time. In the total solution time the gain in time is quite considerable. When we add up all the execution time to solve the first stage problems, we obtain a value of 210,04 s, a clear improvement compared to the previous strategy where we

obtained a value of 975,65 s. Now the time to solve the first stage problems is smaller than the time to solve the second stage problems : 77,74 s.

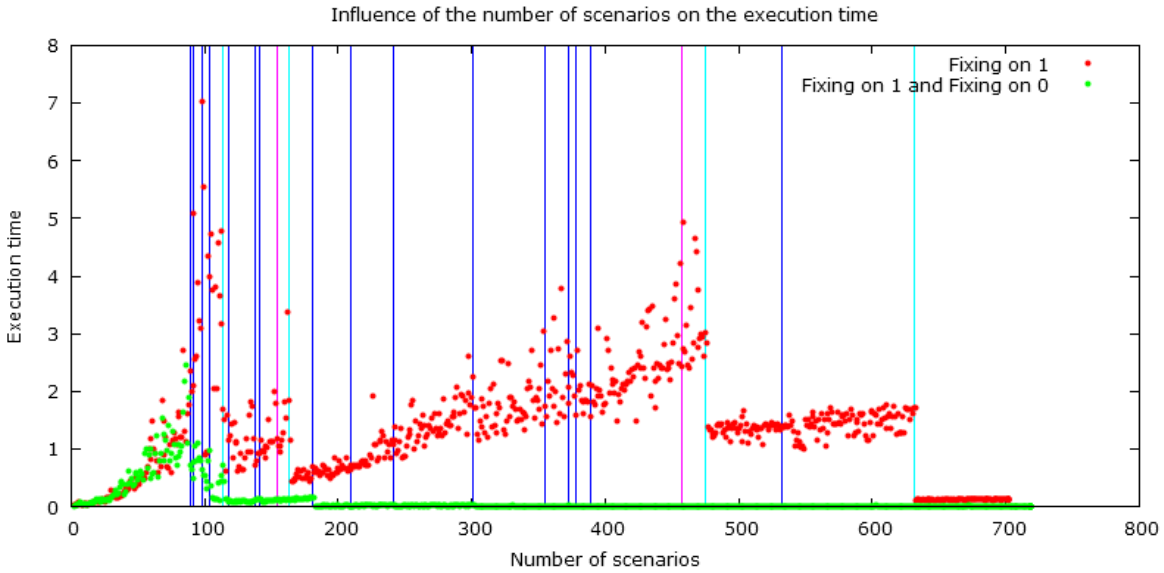

Figure 4.10: Two-phase approach with fixing variables on 1 and 0

Removing scenarios out of the scenario set

Another measure we applied to gain even more time is to remove the scenarios corresponding to the fixed variables. We decided that every variable that we fix at zero is a variable to which the problem is not sensible. This means that the scenarios in which the variable takes other values than it's nominal value are no longer interesting for finding the optimal solution. Now we can remove these scenarios out of the scenario set on which the problem is solved.

An inconvenient of this last measure is that removing scenarios may bring the scenario set out of balance. For example if we have 3 remaining solutions and all solutions have 2 common variables, but there are also 2 solutions that have 2 additional common variables. If we fix a non-common variable to zero and one of the two solutions having 4 common variables becomes unfeasible, we will remove all scenarios using this variable out of the scenario set. Now indirectly a lot of scenarios to which the other solution having 4 common variables was sensible disappeared of the scenario set. This solution will now become more attractive than the third solution and thus for some iterations this solution will be found as the optimal solution. After a number of iterations enough bad scenarios for this solution will have been added, that the scenario set is balanced again and all solutions are attractive as before the variable was fixed. This situation is also graphically explained in figure 4.11 with 10 scenarios. During the recovery period we won't change the scores of variables.

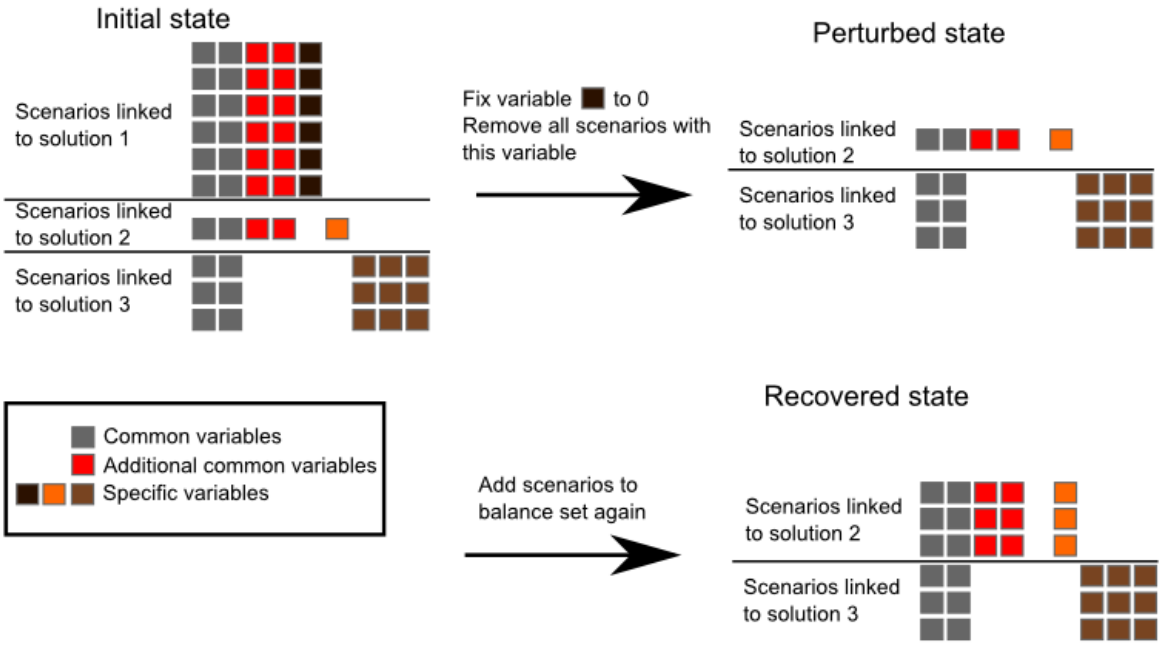

Figure 4.11: Balancing of the scenario set

The gain in time this measure results in is depicted in figure 4.12. Graphically the difference is not clear between the two approaches. If we compare the total time to solve the first stage problems, we obtain 184,24 s for the new method compared to the 210,04 s found above

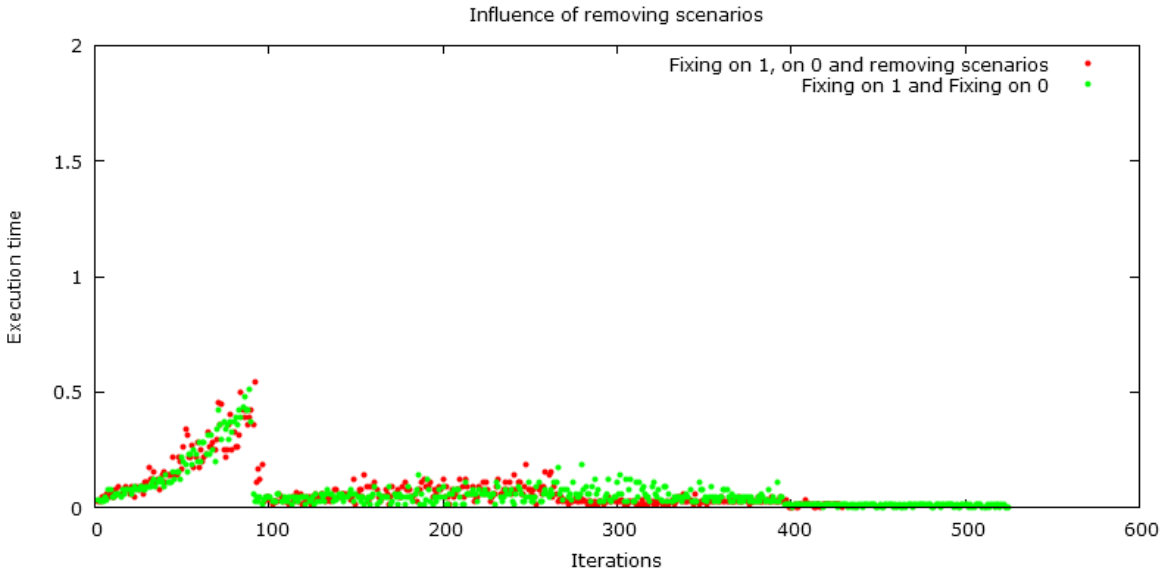

Figure 4.12: Two-phase approach with fixing variables on 1 and 0 and removing scenarios

Differentiating initial scores

The next improvement we are going to introduce is the differentiation in initial score of the transportation variables. The idea is that some variables are in general more interesting to appear in a solution than others. We will use two criteria to change the initial score of a variable. The first criterion is the distance between two locations i and j corresponding to the transportation variable x_{ij}. The bigger the distance between two location the lower the initial score will be set. For the second criterion we will look for locations that are approximately on one line. If we find three or more location approximately on the same line, it would be logic that a tour including all three will pass by one extreme point, then the middle point and then the other extreme. The road between the two extremes will rarely be used. We assume here that the triangle inequality is satisfied(which is a valid assumption for our instance set, but this may not be the case for a general instance set). Examples of both criteria are given in figure 4.13. The blue point is the point around which the distances are measured for the distance based criteria. For the line based criteria the blue point represents the location in between two extremes.

Since we have the distance matrix, the first criterion can easily be applied. The second criterion is a bit more tricky. We found a technique often used in image analysis called the Hough transform[2] which is used to identify instances of objects within a certain class of shapes. Without going into the technical details of the Hough transform, we will give an brief explanation of how the methods works for detecting straight lines.

In a two dimensional space we can determine every point by two coordinates x and y and every line by an intercept b and a slope m using the carthesian coordinate system: $y = mx+b$. If we make the transformation to a polar coordinate system, we use a radius r and a angle θ to define the coordinates x and y as r $\cos(\theta)$ and r $\sin(\theta)$. The line is determined as

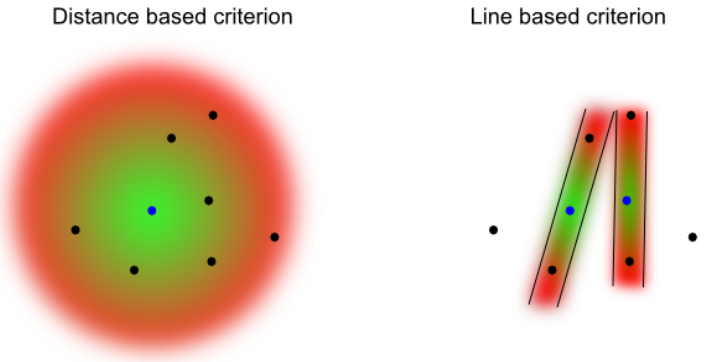

Figure 4.13: criteria to differentiate inital scores

$y = \frac{-cos(\theta)}{sin(\theta)}x + \frac{r}{sin(\theta)}$ or after rewriting : $r = cos(\theta)x + sin(\theta)y$. By varying θ between 0 and 2 π we will obtain lines r(θ) for every point. Wherever the lines of multiple points cross, the points are colinear. For our purpose it will satisfy to find regions where the points are sufficiently close. This technique is shown is figure 4.14 for an instance with 5 clients.

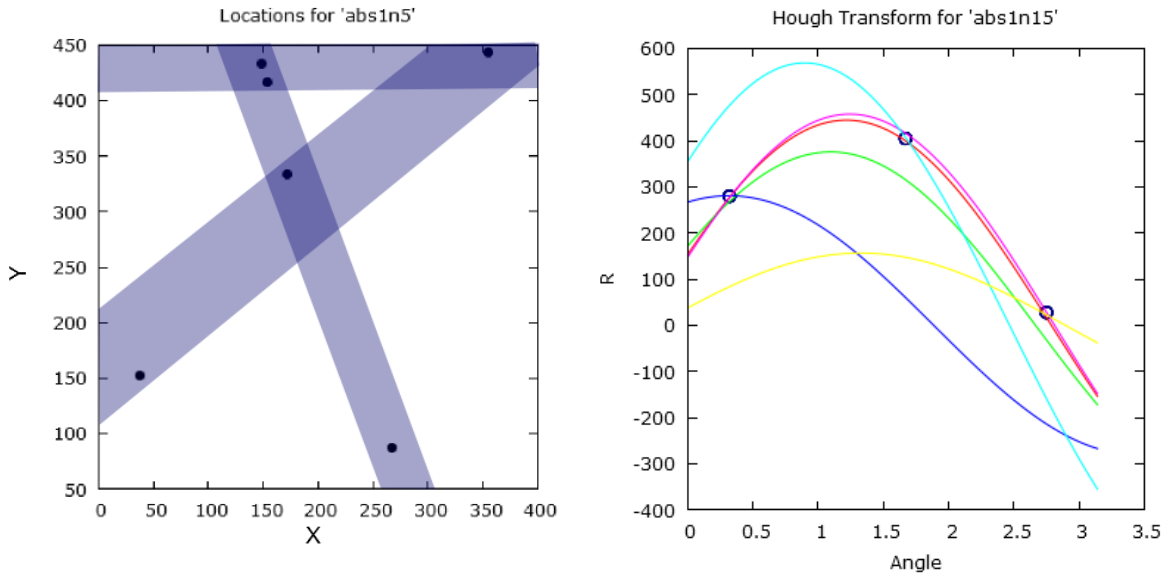

Figure 4.14: Example of Hough transform

We defined the initial scores based on the distances as $\frac{100}{d_{ij}}$. Our instance set places clients in a 500 x 500 grid, so the minimum score to start is 0.14(still 52 iterations away from the critical bound). We will multiply this number by 0,95 if we find a line on which location i and j are situated as two extremes. This corresponds with reducing the fixation of a variable with one extra iteration. The execution time of the first stage problem is depicted in figure 4.15. We can clearly see how the new improvements slow down the initial exponential growth in the beginning. The total time it takes to solve all the first stage problems is now 156,37 s.

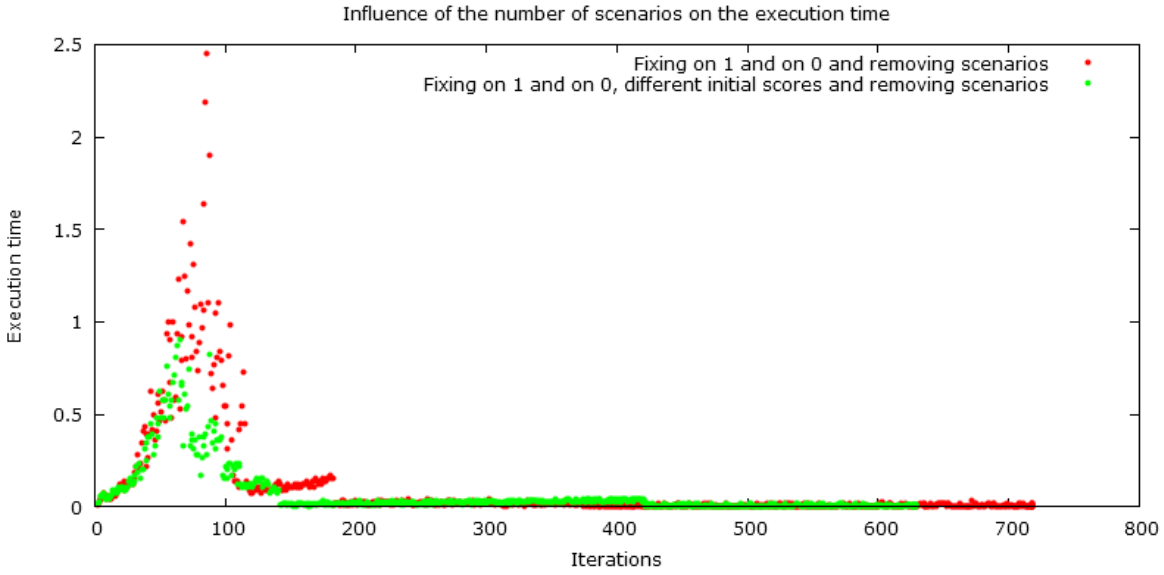

Figure 4.15: Two-phase approach with fixing variables on 1 and 0 and different initial scores

Cutting the tail

As a final reduction in time we will cut the tail of the procedure. After a number of iterations most or all variables will be fixed and we will always find the same solution. At this point the method has converged towards one solution. When we are in this situation, we can stop the procedure. To determine when to stop the procedure we will keep a variable and change it, analogue to the scores of the transport variables in the fixation of the zeros. Every time the same solution is found, we will multiply the variable with 0,95 and every time another solution is found we will multiply it by 2:

$$score_{ij}^{t} = \begin{cases} \max\{2\ score;\ 1\} & \text{if the same solution is found} \\ 0,95\ score_{ij}^{t} & \text{otherwise} \end{cases}$$

4.4.3 Interaction between first and second stage

In the previous sections we have been somewhat vague about how many scenarios to evaluate in the second stage. Initially we took 1000 scenarios. For the further improvements we took 100 scenarios. The reason to take initially this many scenarios is to have scenarios near the worst scenario. This would probably help the convergence of the method and so we can avoid the long execution times in the later iterations. For the improvements we took 100 scenarios because this is still a large number and it is likely that the same scenarios are found in approximately the same order in two runs which allows us to compare results and evaluate improvements. In the beginning of the improvements the exection time of the second stage

was also neglectable compared to the execution time of the first stage. This is no longer the case. We will now evaluate the case with one scenario in the second stage.

This has two effects on the execution time. First the execution time of the second stage decreases, since there are less scenarios to evaluate. Second this will also decrease the time of the first stage. Although it seems like the two stages are independent, this is not really true. Evaluating less scenarios in the second stage will result in the scenarios being more alike. The more alike the scenarios are, the easier the first stage problem is to solve. Decreasing the number of scenarios to be evaluated affects thus both stages. The effect can be seen in figure 4.16.

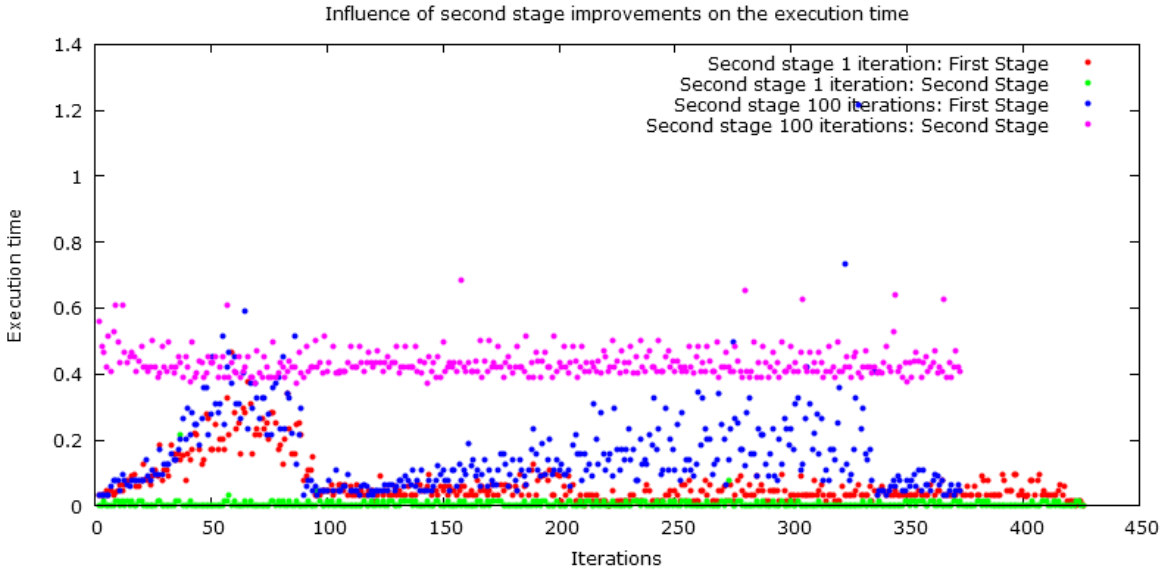

Figure 4.16: Effect of reducing the number of iterations in the second stage problem

After all these improvements it takes between 150 s and 250 s to run the approach on the instance 'abs1n5'. This is a huge reduction of time compared to the 65144,36 s, we needed initally to apply the approach to this instance.

4.4.4 Changing the first stage problem

After the focus on time improvement, we will now evaluate the accuracy of the method. Before testing the accuracy of the method for specific examples we can already make some general comments. Notice that in terms of evaluation of accuracy this approach will differ a lot from the first two approaches. There's a big chance that two runs of this approach will be different, due to the randomness involved in solving the second stage problems. In the first two approaches the results will always be exactly the same for two runs. Next to the randomness

in the method, we also added some measures to speed up the procedure: fixing variables, removing scenarios and giving initial scores to variables. If we change the parameters(for example the increase and decrease functions of the score) of these measures, the method may produce more or less accurate results. Also increasing the number of discretization levels or increasing the number of iterations for the second stage problem may change the accuracy. The accuracy depends thus on a number of parameters and configurations.

Here we will evaluate the accuracy for the parameters and configurations described in the section above. If we apply the method 100 times to the instance 'abs1n5' we get the results shown in table 4.4. Knowing that the optimal solution has a maximum tour length of 907, we see that the optimal solution is found in 75 cases. We can see that there are 25 runs that find a different solution as optimal. We may say that this method works quite good for this particular instance, obviously not perfectly since we find 1 fourth of the time a suboptimal solution.

Longest Tour of the Solution	907	911	932	1141
Frequency	75	17	3	5

Table 4.4: Evaluation accuracy two-phase method for instance with 5 clients

However, when applying the method to other instances, we find that the results are not always as positive as in the last case. For example applying the method to the instance 'abs1n10' always results in finding the nominal solution(corresponding with a tour length of 1346). In table 4.5 the results of the Monte Carlo simulation of the naive approach are shown. We can clearly see that the solution with a tour length of 1286 has on average the lowest cost. If we apply Monte Carlo simulation for the discretization levels we used we can see that between the different solutions is narrowing. The reason why we obtain such different results is that the gap between the neutral scenario(all coefficients take the nominal value) and the worst scenarios is very big. If we bring in mind, that the two-phase approach stops after a number of times the same solution is found, it is probable that the distinction between the solutions is not large enough when the procedure has already stopped.

Longest tour in the solution	1346	1286	1261	1237	1229	1210
Monte Carlo - normal	4760,08	4705,87	4730,55	4831,29	4966,33	4993,21
Monte Carlo - discretization levels	4760,20	4731,21	4763,86	4869,31	5006,22	5035,76

Table 4.5: Difference between Monte Carlo applied under normal circumstances and under discretization levels

We see two solutions for this problem. The first one is increasing the number of discretization levels. An inconvenient of this solution is the enormous increase in possible scenarios. This would also have consequences on the measures we took to speed up the procedure. We like to start fixing variables as soon as possible to avoid the exponential increase and the first scenarios found are the scenarios situated at the extremes of the hypercube. The scenarios taking into account the new discretization levels would thus be found later in the procedure and would thus have no influence on the important decisions taken in the beginning. We could start our measures later in the procedure but this would increase the execution time drastically. Because of these reasons, increasing the number of discretization levels is not that interesting. An other solution is altering the formulation of the first stage problem. Instead of taking the scenarios exactly as found, we could alter the scenarios slightly in a way that they keep part of their caracteristics but are not as extreme as before. This could be done by taking part of their values c_{ij}^p and adding part of the values of the neutral scenario c_{ij}^n:

$$c_{ij} = \alpha\ c_{ij}^p + (1-\alpha)\ c_{ij}^n$$

In this notation α is a random variable that takes values in the interval $[0;1]$. Notice that this approach is related to increasing the number of discretization levels. Due to our new altered scenarios the gap between the neutral scenario and the worst case scenario is reduced. The first stage problem now changes:

$$min \sum_{i\in V}\sum_{t\in T} h_i I_i^t + \sum_{p\in S}\sum_{i\in V}\sum_{j\in V}\sum_{t\in T} \frac{\alpha\ c_{ij}^p + (1-\alpha)\ c_{ij}^n}{|S|} x_{ij}^t + \sum_{p\in S}\sum_{t\in T} \frac{Penalty}{|S|} P_t^p \quad (1)$$

subject to $(2)-(15)$

$$\sum_{i\in V}\sum_{j\in V} (\alpha\ c_{ij}^p + (1-\alpha)\ c_{ij}^n) x_{ij}^t \leq T + M P_t^p\ \forall t \in T,\ \forall p \in S \quad (16)$$

$$P_t^p \in \{0,1\}\ \forall t \in T,\ \forall p \in S \quad (17)$$

If we now apply this adapted procedure to the instance 'abs1n10', we observe a higher accuracy. Further tests will be evaluated in the section about computational results.

4.4.5 Conclusion

As a final conclusion we can state that this method is suited for both small and larger instances. Because of our improvements we are not as much confronted with the exponential increase of time. On the other hand the fixation of variables has to happen delicately so that no interesting solutions are eliminated. Using the second stage to introduce new scenarios seems to work rather well. We are however confronted with the time this adds to the method.

Therefore, we will try in following section to eliminate this stage and look at the results this gives. A drawback of this method is that it is specific for every configuration of *TimeLimit*, *Penalty* and maximum deviation. Every change in configuration demands thus a new run of the approach.

4.5 Scenario Approach

Before analysing the scenario approach in terms of calculation time and accuracy, we want to emphasize that the scenario approach can be applied in different ways. A first way to apply the scenario approach is to solve the variable IRP for a fixed number of scenarios. The number of scenarios will be a crucial element in this approach. If the number of scenarios is too small, the set S may not cover the full range of the uncertainty and solution of the variable IRP based on this small set can be different for various executions of the algorithm(and thus different from the optimal solution if more scenarios are included). This situation is depicted in figure 4.17 on the left. The optimal solution in this example corresponds with a maximum tour length of 907. We run the scenario approach 100 times. As we can see the accuracy of the approach is very low for a set with 100 scenarios. If we increase the number of scenarios to 500, depicted in figure 4.17 on the right, the accuracy increases but slowly. Taking 5 times more scenarios increases the accuracy only with 17,3% from 48,9% to 66,2%. Remind that all scenarios are generated randomly.

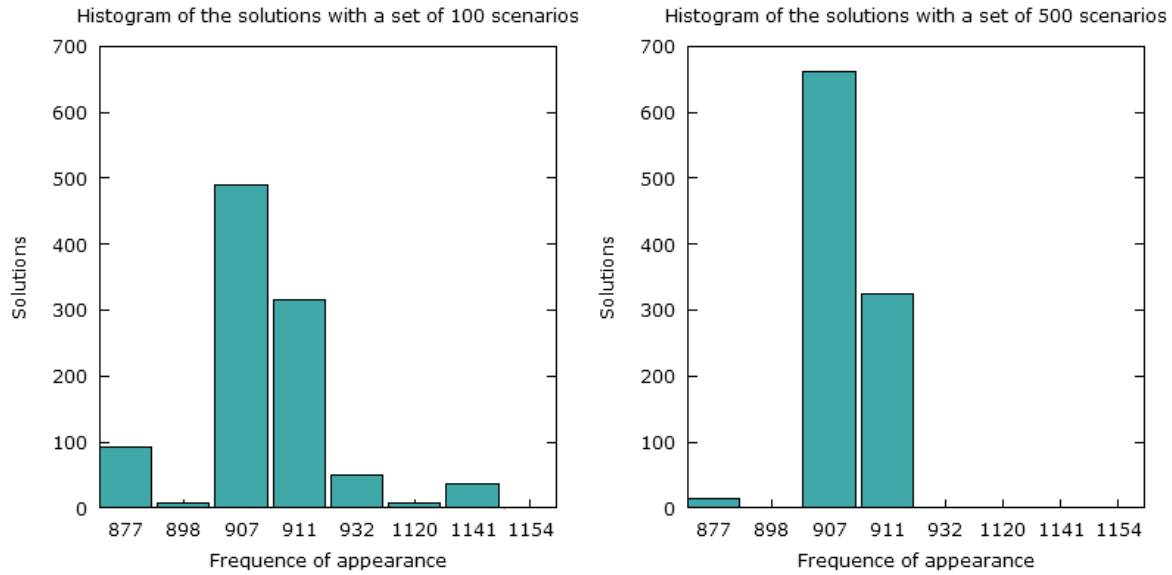

Figure 4.17: Histogram of the solutions with a set of 100 scenarios and a set of 500 scenarios

This increase in the number of scenarios has also its cost on the solution time of the problem.

Increasing the number of scenarios means an increase in constraints and variables. For this problem the solution time is exponential in function of the number of scenarios. This can be seen in figure 4.18 in which the solution time is shown in function of the number of scenarios.

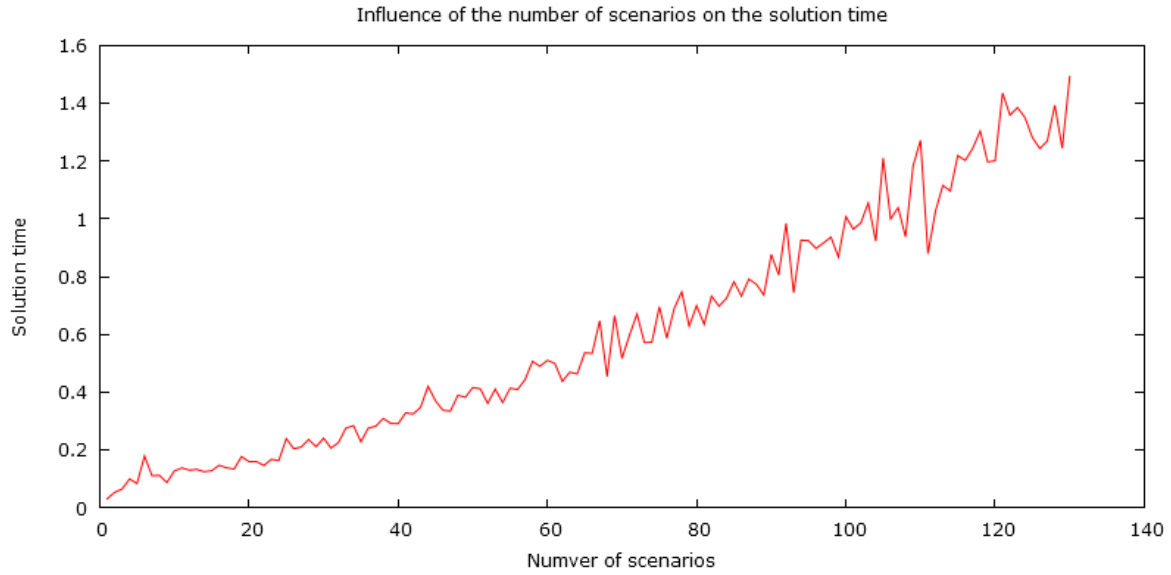

Figure 4.18: Solution time in function of number of scenarios

A better approach might be to have the number of scenarios variable, like in the two-phase approach. We can start with an initial number of scenarios and solve the problem. In the next iteration we increase the number of scenarios and solve the problem again. Through various iterations we will converge to the optimal solution and finally have a sufficient number of scenarios to identify with a high probability the optimal solution. Note that the problem of the increasing complexity(and time) isn't solved, but as we saw in the section on the two-phase approach there are methods to deal with this problem. This method is shown in figure 4.19. The horizontal axis shows the number of scenarios and the vertical axis shows the longest tour associated with the solution found. We remark the variation in the beginning but eventually, we converge to the value of 907.

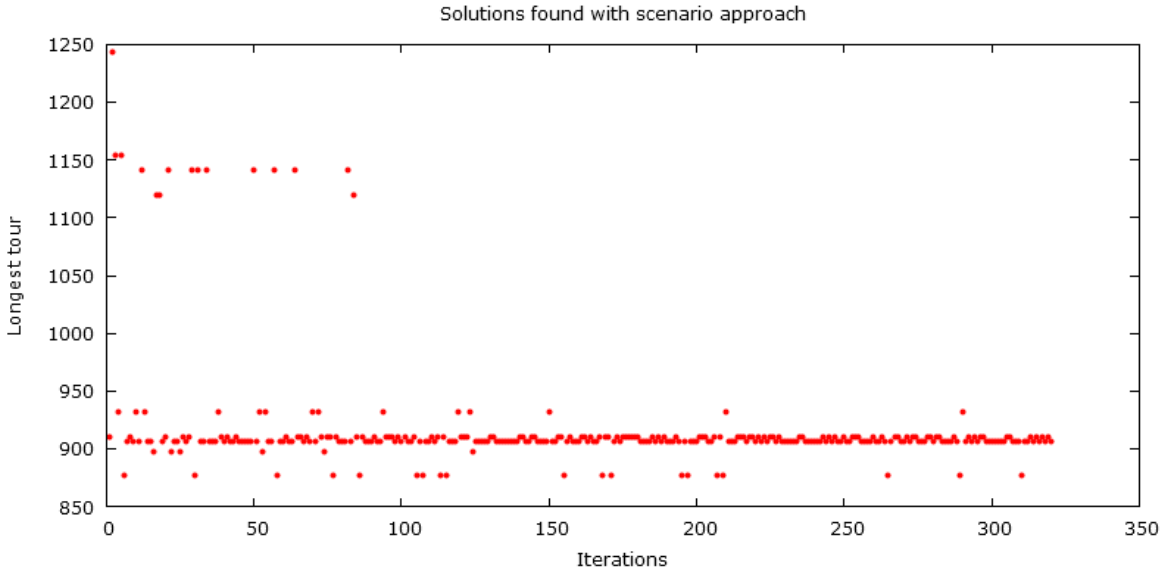

Figure 4.19: Scenario approach for the instance 'abs1n5'

4.5.1 Antithetic variates

Before applying all the improvements of the last section we will first introduce one measure specific for this approach. The idea is based on antithetic variates. In statistics antithetic variates are normally used to reduce the variation of a Monte Carlo method. The idea is to generate for every sample another sample that is negatively correlated with this sample. As a result of this negative correlation the variance of the global estimator decreases. Our case is not the classical application of antithetic variates, but using this idea ensures that our scenario set S will be balanced. The more balanced our set is, the faster the method will converge and the faster the method will terminate.

Since the scenarios are generated randomly from a uniform distribution, we can construct from every sample with coefficients $\alpha * c_{ij}^n$, in which c_{ij}^n is the nominal value of the coefficient and α is taken in the range $[1 - MD, 1 + MD]$ (MD=maximum deviation), another sample with coefficients $(2 - \alpha) * c_{ij}^n$. This will always produce negatively correlated samples. For example if α equals 1-MD, the first sample will have coefficients $(1 - MD) * c_{ij}^n$ and the second sample will have coefficients $(1 + MD) * c_{ij}^n$ The effect on the convergence of the method by using antithetic variates can be seen in figure 4.20. Note that we take 2 samples of the original method for every sample generated with the antithetic variates, so that the total number of scenarios is the same.

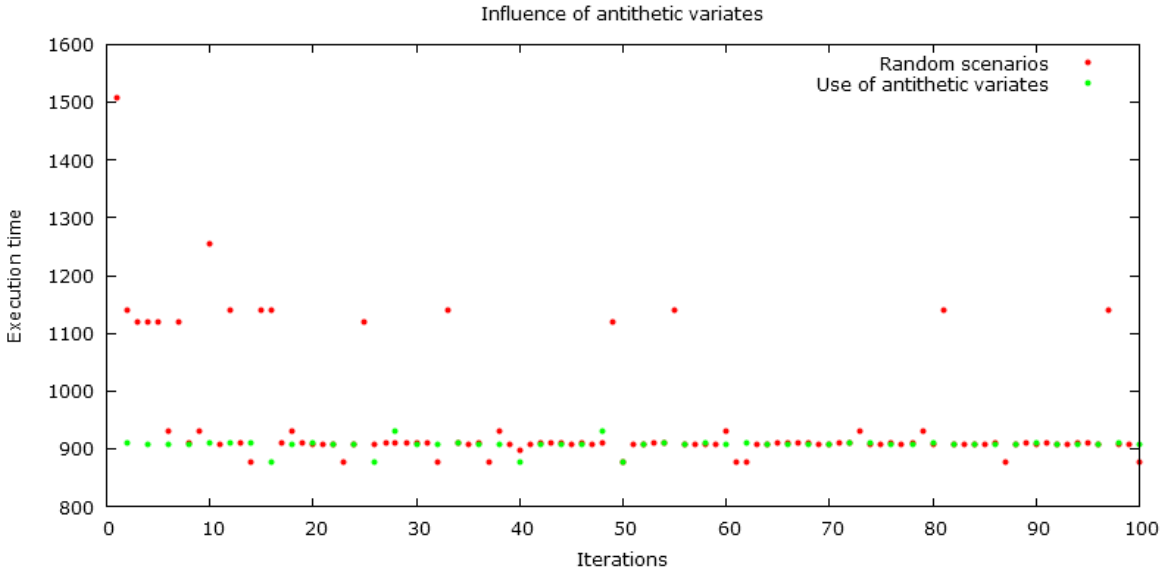

Figure 4.20: Influence of antithetic variates

4.5.2 Improvements Two-Phase approach

We can now apply the improvements we used in the two-phase approach: fixing variables on 1, fixing variables on 0, changing initial scores and cutting the tail. As the explanation of these improvements is similar to the one given in the section above about the two-phase approach, we will just give the results of applying the various improvements. For the discussion we refer to the previous section. In figure 4.21 the effects of the improvements are shown. The final time it takes to solve the instance 'abs1n5' is 8,57 s.

Figure 4.21: Influence of all the improvements

4.5.3 Evaluation of accuracy

Since we adon't have fixed discretization levels, we are not facing the problem we had in the two-phase approach. Furthermore the faster convergence, due to the antithetic variates, makes it less likely to end up with a wrong solution. We will further compare the accuracy of the method with the other approaches in section 5.

4.5.4 Conclusion

Similarly to the two-phase approach we can conclude that this method is suitable for both large and small instances. Dropping the second stage has a huge improvement on the run time of the method. A drawback of this method is that the distribution of the travel times is assumed to be known. If not, the improvement of the antithetic variates cannot be applied.

4.6 Estimating the cost

In the previous sections we have always focused on finding the most optimal solution. Nonetheless, it is also important to know or be at least able to estimate the cost distribution associated with this solution. An unconvenient of the first two approaches, the naive and robust approach, is that the cost distribution can only be determined by Monte Carlo simulation. The two latter approaches, two-phase approach and scenario approach provide already estimates of the cost distribution before Monte Carlo simulation. At the end of both approaches the

value of the objective function is a rough estimate of the average cost of the solution. Furtermore, by solving the problems of the second stage we can estimate the minimum cost and the maximum cost. So now we can already roughly estimate the range of the cost.

But we can go further. Since the travel times are assumed to be uniformly distributed in a bounded symmetric interval around their nominal value, we know their expected value and variance. Based on this information we can for each period calculate the average and the variance of the cost distribution(we will only look at the transport costs, since holding costs are fixed):

$$E[\sum_{i\in V}\sum_{j\in V} c_{ij}x_{ij}^t] = \sum_{i\in V}\sum_{j\in V} E[c_{ij}x_{ij}^t] = \sum_{i\in V}\sum_{j\in V} x_{ij}^t E[c_{ij}] = \sum_{i\in V}\sum_{j\in V} x_{ij}^t \tfrac{c_{ij,max}+c_{ij,min}}{2} = \sum_{i\in V}\sum_{j\in V} x_{ij}^t c_{ij}^n$$

$$Var[\sum_{i\in V}\sum_{j\in V} c_{ij}x_{ij}^t] = \sum_{i\in V}\sum_{j\in V} Var[c_{ij}x_{ij}^t] + \sum_{(i,j)\neq(k,l)} Cov[c_{ij}x_{ij}^t, c_{kl}x_{kl}^t] = \sum_{i\in V}\sum_{j\in V} Var[c_{ij}x_{ij}^t] = \sum_{i\in V}\sum_{j\in V} x_{ij}^t Var[c_{ij}] = \sum_{i\in V}\sum_{j\in V} x_{ij}^t \tfrac{(c_{ij,max}-c_{ij,min})^2}{12}$$

For the calculation of the expected value we use the following properties: $E[\sum X_i] = \sum E[X_i]$ and $E[aX] = aE[X]$. Furthermore we now that the expected value of a continuous uniformly distributed variable equals $\frac{c_{ij,max}+c_{ij,min}}{2}$. Since the travel times are uniformly distributed in a bounded symmetric interval around their nominal value, $\frac{c_{ij,max}+c_{ij,min}}{2} = c_{ij}^n$.

For the calculation of the variance we used the following properties: $Var[\sum X_i] = \sum Var[X_i] + \sum_{i\neq j} Cov[X_i, X_j]$ and $Var[aX] = a^2E[X]$. All $Cov[X_i, X_j]$ equal zero, since the length of one road is independent of the length of the others. Therefore, we can drop this term in our calculation. Furter we know that x_{ij}^t takes either the value 0 or 1, so ${x_{ij}^t}^2 = x_{ij}^t$. Finally the variance of a continuous uniformly distributed variable equals $\frac{(c_{ij,max}-c_{ij,min})^2}{12}$.

Now we can distinguish two cases. We will first look at the case in which all deliveries are done in one period. In this case the cost of the solution will consist of the fixed costs related to the inventory, the variable transportation costs of the one period and the penalty that may or may not have to be paid. The cost distribution will consist of 2 parts. The first part of the distribution will have a normal distribution with mean the optimal objective value of the nominal problem and standard variation $\sum_{i\in V}\sum_{j\in V} x_{ij}^t \frac{(c_{ij,max}-c_{ij,min})^2}{12}$ from the minimum possible value of the tour, so putting all values c_{ij} at their lowest possible value $c_{ij,min}$, to the $TimeLimit$ we fixed. The second part will have a normal distribution with mean the optimal objective value of the nominal problem and the penalty to be paid and a standard variation of $\sum_{i\in V}\sum_{j\in V} x_{ij}^t \frac{(c_{ij,max}-c_{ij,min})^2}{12}$. This part will go from $TimeLimit + Penalty$ to the maximum

possible value of the tour, $c_{ij,max}$, and the penalty. To include the fixed inventory costs we will shift the whole distribution over the value of the fixed inventory costs.

An example of this situation is shown in figure 4.22. In red a histogram of the cost distribution is shown with Monte Carlo simulation. In black the distribution is shown as discribed above. In this case the $TimeLimit$ was chosen as the largest tour of the nominal IRP, a value of 1141,0. The holding costs are 967,34. As described above the first part of the cost distribution is situated between 1537,84, the minimum value, and 2108,34. This part follows a normal distribution with mean 2108,34 and standard deviation 156,39. The penalty is 500 in this example, so the second part of the cost distribution is situated between 2608,34 and 3178,84.

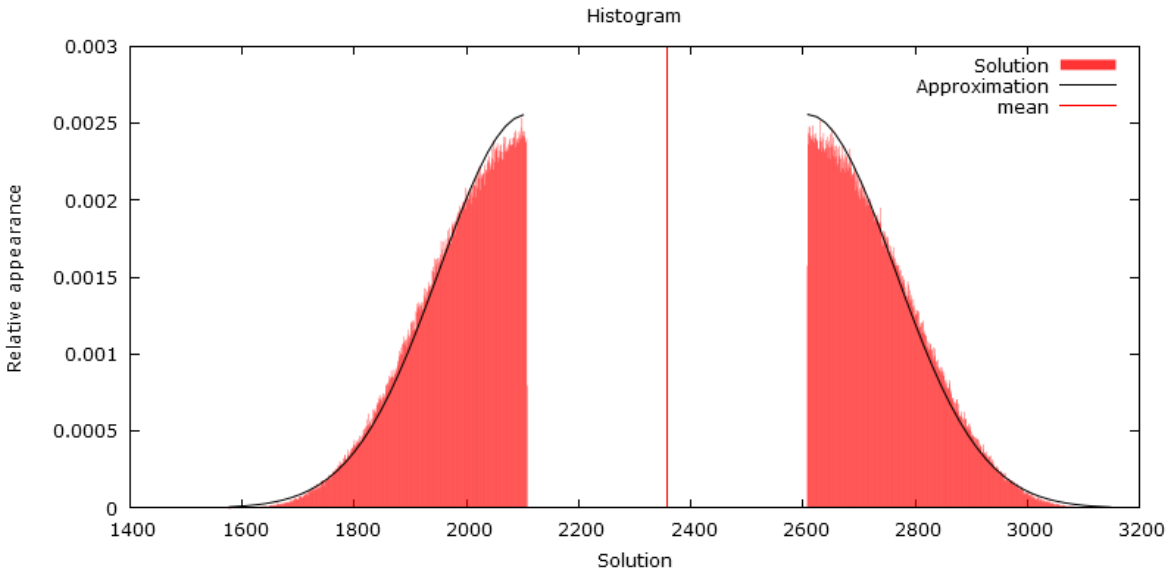

Figure 4.22: Simple cost distribution

If the deliveries are done in different periods the estimation of the cost is more complicated. The cost distributions of one tour will effect the cost distributions of the other tours and there may be multiple tours that can violate the $TimeLimit$. The cost distribution gets thus more complicated. An example of such a complicated cost distribution is given in figure 4.23. Note the the simple distribution described above is no longer a good approximation of the real cost. However, to obtain a good approximation more complicated mathematical concepts are required such as convolutions of partial distributions. A full analysis of the cost distribution is out of the scope of this master dissertation. Nonetheless the approximation above still gives us some intuition on how the real cost is distributed

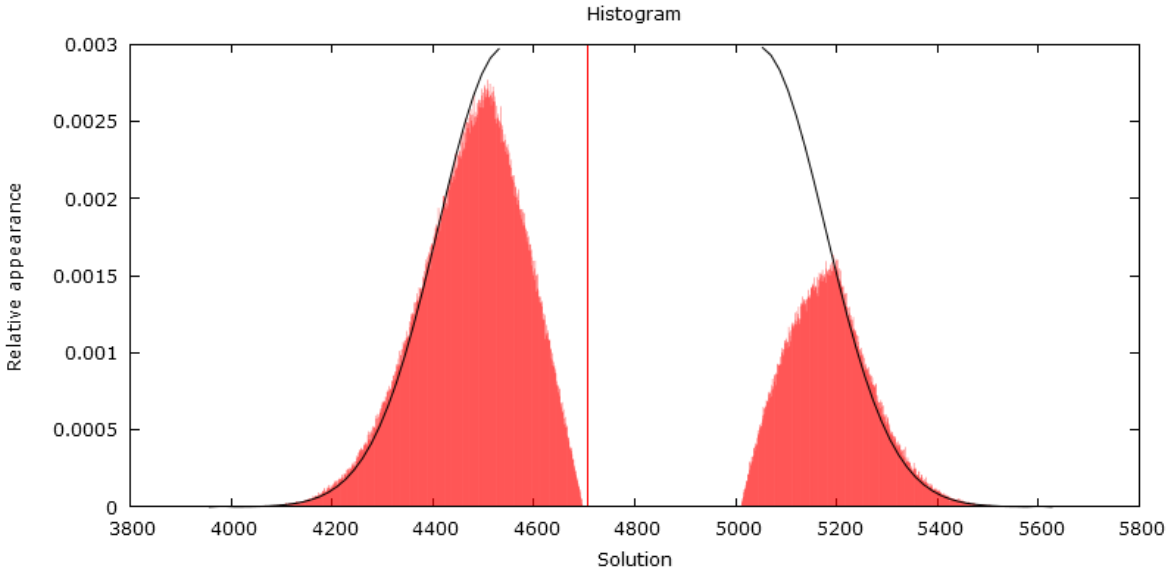

Figure 4.23: Complicated cost distribution

Chapter 5

Computational results

This chapter discusses the results of experiments conducted for the four different models on a dataset in which the number of clients varies between 5 and 50 clients and the *Penalty* is fixed on 500 or 1000, dependent on the number of clients. The problems were solved with CPLEX 12.6. The experiments were all executed on Dell PowerEdge 1950 Server with 4 E5420 Xeon Dual Processors. All the results can be consulted in the Appendix. For the naive approach and the robust approach the procedure was executed once. For the two-phase approach and the scenario approach the procedure was executed 50 times for the instances up to 20 clients. In this way the average behaviour for these instances could be estimated. For the other instances we only ran the procedure once. The reason is that we observed very long execution times.

In table 1 and 2 the execution times that the different approaches needed in order to solve the problems are depicted. In table 1 the number of accurate iterations for the naive approach and the robust approach is also indicated. We define an accuracte iteration if the execution time doesn't exceed 600 s. If it does exceed this time limit, the current best solution is given without proof of optimality. In table 2 the standard deviation of the execution times is also given.

In table 3 and 4 the accuracy of the method is given. For the naive approach and the robust approach, we evaluate the accuracy by checking if the optimal solution appears in the front of the naive approach. Next we calculate the average deviation between the best solution found and the optimal solution. If the optimal solution appears in the front, the deviation will thus be 0,00. For the two phase approach and the scenario approach, we calculate how many times the optimal solution of naive approach is found at the end of the procedure. Next we calculate the average deviation between all solutions found and the optimal solution. Negative deviations will appear if the method finds a better solution than the ones in the front of the naive approach.

Finally we remark that the solutions times may be different from the solution times previously stated. The reason is that the experiments in the previous section were conducted on a different machine.

5.1 Execution time

If we look at the execution times in table 1 we can see that for the naive approach and robust approach the execution time is increasing with the number of clients for the first 5 instances. For the last 5 instances this increase is not so clear anymore. We notice also that in these instances the front has more inaccurate iterations than in the first 5 instances. Since the inaccurate iterations are stopped after the time limit of 600 s. It is thus probable that the increase in execution time is also valid for the last 5 instances but that the time limit on the iterations masks this effect. If we compare the naive approach and robust approach, we see that there is not a clear distinction in execution time between the two approaches. For the bigger instances we can see shorter times for the robust approach. However, this is due to the lower number of iterations of the robust approach and the high percentage of inaccurate iterations. Based on these results, we can thus not favor one method over the other in terms of execution time. We feel, based on our observations in the section 4.3.2, that removing the time limit would enable us to make the distinction between the naive approach and robust approach and that this would show that the naive approach is faster than the robust approach.

If we look at the execution times for the two phase approach and the scenario approach in table 2, we can see that for the scenario approach we need less time, except for the instance with 20 clients. We can also see that for the first 4 instances the scenario approach has a smaller standard deviation. Differences between 2 runs are thus not as big as in the two phase approach. We can also clearly see the increase in execution time for the 4 first instances of the scenario approach. For the instances with more clients this increase dissappears a bit. The reason is that the measures we took to determine the initial scores worked very good for the small instances, but lower the scores too much for the large instances. At the start of the procedure already a lot of scores are fixed at 0 and thus a lot of solutions are a priori eliminated. The difference between the solutions that remain possible is quite large and thus stops the procedure after a relative low number of iterations. In the two phase approach we observe the same phenomenon(the a priori elimination of a lot of solutions) but this does not lead to short execution times. The difference here with the scenario approach is that the scenarios chosen in the two phase approach are more solution-specific. As explained in the section on the two phase approach generating scenarios for one solution makes in general other solutions scenarios more attractive. Even though the difference between two solutions may be large, eventually such a solution may be evaluated in the two phase approach. In the scenario approach the scenarios are generated randomly and they have the same effect on all

solutions. So in the scenario approach we are not confronted with this effect.

If we compare the execution time of all methods, we can see that for small instances the first two methods are much faster than the second two. For large instances we observe that the most time consuming method is the two phase approach. For the other methods it is hard to make a distinction since we have iterations that are all bounded by the time limit.

5.2 Accuracy

In table 3 the results in terms of accuracy are given for the naive and the robust approach. For the robust approach we see that the optimal solution doesn't always appear in the front of the naive approach. However, the deviation of the optimal solution is not really large. Even for the large instances the robust approach still finds solutions that are relatively close to the optimal solution.

Looking at the results for the two phase approach and scenario approach in table 4, we can conclude that the methods work quite accurate for small instances. For the instance with 15 clients we see that the scenario approach only finds the current optimal solution in 34 of the 50 executions. In the other executions a solutions is found that has a cost extremely close to the optimal cost(less than 0,001 %). We notice that the accuracy decreases severely if the number of clients increase. The reason behind is the effect of the a priori elimination of many solutions. Due to this practice a lot of good solutions are already eliminated at the beginning of the procedure. Next we note that the deviation is alsways bigger for the two phase approach than for the scenario approach. This strokes with our findings in the section on accuracy of the two phase approach and the scenario approach. We do bring in mind that the large instances only have been solved once, so that this result only gives an indication. For the last instance the deviation is the same, since so many solutions are eliminated that both procedures jump to the same solution immediately.

5.3 Dropping heuristic

Because our findings on accuracy are a bit perturbed by the differentiation of the initial scores, we also executed both procedures on the instances without the measures to change initial scores(only one time). The results can be seen in table 5 and table 6. Regarding the time, we can say that we still have a steep increase in execution time when increasing the number of clients. Compared to table 2 we obtain in general longer execution times. This is logic since we removed one of the improvements that speed up the procedure. Looking at table 6 we see that the performance in terms of accuracy increases for both methods. For the

scenario approach we find even better solutions, resulting in a negative deviation compared to the optimal cost. For these large instances the scenario approach finds thus solutions that are not appearing in the front of the naive approach.

5.4 Conclusion

We can conclude that for small instances(5-15 clients) the naive approach, the two phase approach and the scenario approach give similar results in terms of accuracy. In execution time the naive approach outperforms the other two methods. For small instances the naive approach is thus the better of the four methods. For large instances($\geq$ 20 clients) there is not one method that dominates the others in both criteria. If execution time is the most important criterion the naive approach and the robust approach would be good methods since they cover the whole range of variability and we saw that for our dataset they performed well in terms of accuracy. They did not attain the optimum that was attained by the scenario approach with equal inital scores, but the deviation is limited, $\leq 2\%$. If accuracy is the most important criterion, the scenario approach outperforms the other methods.

Chapter 6

Conclusions

In this master dissertation we have introduced the notion of variable travel times in the classical IRP. We explored different ways of modelling the problem. In each of our approaches we proposed measures to improve the speed and the accuracy of the method. In the next paragraphs we outline our main findings as well as suggestions for future research.

We have proposed a brief literature review in Chapter 2. A standard model of the inventory-routing problem was presented. Furthermore the relevant literature that adresses variability in the IRP and related problems was reviewed. Based on an article of Solyali et al.[28] that used robust optimization to solve the IRP with stochastic demand and showed positive results, a brief overview of the main methods in robust optimization was also presented.

In Chapter 3 the dataset for the IRP we used, was first examined for its sentivity to variability in the travel times. To make the instances more sensitive to changes in travel times, the model was extended with a time limit constraint. Then we presented 4 different approaches to adress the variability on the travel times in the IRP. In the first two approaches, the naive approach and the robust approach, the general strategy was to identify the interesting solutions over the range of variability and in this way construct a front of good solutions. Afterwards the best solution in the front was found by appying Monte Carlo simulation. The second two approaches, the two phase approach and the scenario approach, differ greatly from the first two in the way that they don't identify solutions and test them, but solve the problem for a set of scenarios. The solution found is the optimal solution for the IRPVTT. The comparison between solutions happens thus in the procedure itself and not afterwards.

In Chapter 4 we presented methods to speed up the execution time of the different approaches and improve their accuracy. For the first two methods improvements included a proper stopping criterion for both methods. For the last two methods a wide range of improvements were proposed. For the two phase approach first the second phase was simplified. Then fixation of variables along the procedure was proposed. This idea was based on the convergence of the

method. Finally, heuristics were introduced so that variables that are likely to be fixed could be fixed earlier in the procedure. A stopping criterion was also introduced for this method. Most of the improvements of the two phase approach could also be used in the scenario approach. One specific improvement of the scenario approach was the introduction of antithetic variates. For every scenario introduced a negatively correlated scenario was introduced. This accelerated the convergence of the method. In the last part of Chapter 4 we also briefly discussed how to estimate the cost distribution of the IRP with variable travel times without having to apply Monte Carlo simulation. For this estimate the distribution of the travel times has to be known in advance.

Finally, in Chapter 5 we compared the methods based on a data set provided on http://www.leandro-coelho.com/instances/thesis/exact_irp/. We evaluated the execution time and accuracy of the methods. Our main conclusions are that all the approaches show a significant increase in execution time when the instances become larger(i.e. more clients). For small instances results indicated that the naive method was the best method both in terms of accuracy and execution time. For larger instances the naive approach, robust approach and scenario approach are suited. If accuracy is the most important criterion, then the scenario approach outperforms the other methods. We noticed also that our heuristic to determine how fast variables can be fixed was too agressive. Dropping the heuristic led to better results.

We view the main scientific contributions of this master dissertation as the four methods introduced to cope with the variability in the travel times. The classical methods to solve problems in mathematical programming, MIP, are not able to solve problems in which the parameters are variable. To tackle such problems the solution methods must be adapted. In this study we presented four such methods. In the development of our methods the focus was both on performance in execution time and accuracy. Execution time is a necessary criterion, since we are dealing with NP-problems and the introduction of variability makes the problem even harder. Accuracy is a good criterion to evaluate the quality of the solution. Our results have showed that the methods presented in this study can solve instances, both small and large, while performing good on both criteria.

We believe that our work opens up to a number of meaningful extensions. First, we chose in our work the simplest relation between travel times and associated costs. However, in reality this relation may be far more complex. It would be good to examine the performance of our solution methods under different relations between travel time and associated costs. Second, we believe that apart from the improvements described in this dissertation there exist a lot of improvements to speed up the two-phase approach and scenario approach. We described two heuristics to differentiate initial scores on which the fixing of variables happens. The results showed that our heuristics were too agressive, but we believe that improvements can be made to increase the performance of these heuristics. Also other heuristics can be explored

to eliminate variables a priori. Another similar improvement we propose is to temporarily fix frequently changing variables. We noticed that there are some variables that are frequently changing in the two-phase approach and scenario approach. By fixing these variables for a number of iterations randomly on one of their possible values, the problem gets easier and the fixation of other variables that are most likely to be fixed will happen thus faster. Afterwards the variables can be 'released' again, but since other variables are fixed now, the problem is easier to solve. Finally, we mention the possibility of combining the different approaches. An example is the combination of the naive method and the scenario approach. If the naive method is applied first, we have a number of possible solutions to our disposal over the range of variability. Afterwards the scenario approach can be applied, but we can use the solutions of the naive approach as upper bounds to eliminate nodes in the branch-and-bound procedure of every iteration and so speed up the whole procedure.

The IRP with variable travel times has shown to be a challenging problem. In this master dissertation we developed different approaches to address the problem. Some of these approaches show a lot of potential and we hope to stimulate other researchers to help developping these approaches further and to pursue the study of this fascinating problem.

Appendix

#clients - *Penalty*	Naive Approach		Robust Approach	
	Time[s]	# Accurate iterations	Time[s]	# Accurate iterations
5-500	0,45	8/8	0,25	3/3
10-500	17,90	13/13	45,88	4/4
15-500	191,55	31/31	78,45	10/10
20-1000	4362,4	9/15	1310,11	3/5
25-1000	2247,76	34/34	3677,36	4/10
30-1000	6081,31	5/14	3636,48	2/8
35-1000	12433,80	12/30	4885,47	2/10
40-1000	7910,75	3/15	2444,79	1/5
45-1000	14742,31	5/28	6057,74	1/11
50-1000	10997,92	1/19	3181,02	1/6

Table 1: Execution time of 10 instances for the naive and the robust approach

#clients - *Penalty*	Two-Phase Approach		Scenario Approach	
	Average Time[s]	Standard Deviation [s]	Average Time[s]	Standard Deviation [s]
5-500	393,44	276,66	13,88	0,99
10-500	222,92	36,28	76,60	3,63
15-500	951,68	214,60	274,76	20,94
20-1000	529,64	229,79	1390,48	159,25
25-1000*	30067,22	-	3903,16	-
30-1000*	18963,83	-	3199,44	-
35-1000*	16605,56	-	4349,28	-
40-1000*	13348,74	-	5623,31	-
45-1000*	71371,32	-	14138,44	-
50-1000*	2129,25	-	1283,90	-

Table 2: Execution time of 10 instances for the two-phase and the scenario approach

#clients - *Penalty*	Naive Approach		Robust Approach	
	Optimal Reached(Y/N)	Average Deviation(%)	Optimal Reached(Y/N)	# Average Deviation(%)
5-500	Y	0,00	N	0,96
10-500	Y	0,00	N	1,16
15-500	Y	0,00	N	0,21
20-1000	Y	0,00	N	0,40
25-1000	Y	0,00	N	0,01
30-1000	Y	0,00	Y	0,00
35-1000	Y	0,00	N	0,01
40-1000	Y	0,00	Y	0,00
45-1000	Y	0,00	N	1,13
50-1000	Y	0,00	N	0,01

Table 3: Accuracy of 10 instances for the naive and the robust approach

#clients - *Penalty*	Two-Phase Approach		Scenario Approach	
	Optimal Reached(%)	Average Deviation(%)	Optimal Reached(%)	Average Deviation(%)
5-500	90,0	0,11	94,0	0,06
10-500	100,0	0,00	100,0	0,00
15-500	100,0	0,00	68,0	0,00
20-1000	0,0	0,43	100,0	0,00
25-1500*	0,0	1,87	100,0	0,00
30-1500*	0,0	1,53	0,0	1,12
35-1500*	0,0	6,90	0,0	4,61
40-1500*	0,0	9,60	0,0	9,02
45-1500*	0,0	15,21	0,0	12,93
50-1500*	0,0	25,10	0,0	25,10

Table 4: Accuracy of 10 instances for the two-phase and the scenario approach

#clients - *Penalty*	Two-Phase Approach	Scenario Approach
	Average Time[s]	Average Time[s]
5-500*	309,24	20,39
10-500*	312,02	136,316
15-500*	1039,51	260,11
20-1000*	531,96	2485,92
25-1000*	22301,24	11136,27
30-1000*	45840,99	16972,44
35-1000*	69597,94	16433,94
40-1000*	51605,78	23344,25
45-1000*	40206,68	22841,68
50-1000*	51245,52	24788,46

Table 5: Execution time of 10 instances for the two-phase and the scenario approach without different initial scores

#clients - *Penalty*	Two-Phase Approach	Scenario Approach
	Average Deviation(%)	Average Deviation(%)
5-500*	0,00	0,00
10-500*	0,00	0,00
15-500*	0,00	0,00
20-1000*	0,43	0,00
25-1000*	0,91	0,00
30-1000*	1,91	-0,23
35-1000*	2,18	-0,68
40-1000*	1,91	-0,12
45-1000*	0,96	-0,27
50-1000*	2,85	-0,29

Table 6: Accuracy of 10 instances for the two-phase and the scenario approach without different initial scores