MULTI-DEPOT VEHICLE SCHEDULING WITH DISRUPTIONS

by

EZGİ YILDIZ

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabancı University
July 2011

MULTI-DEPOT VEHICLE SCHEDULING WITH DISRUPTIONS

APPROVED BY

Assoc. Prof. Ş. İlker Birbil (Thesis Supervisor)	[illegible]
Assist. Prof. Kerem Bülbül	[illegible]
Assoc. Prof. Özgür Gürbüz	[illegible]
Assist. Prof. Güvenç Şahin	[illegible]
Assoc. Prof. Tonguç Ünlüyurt	[illegible]

DATE OF APPROVAL: 26 July 2011

©Ezgi Yıldız 2011
All Rights Reserved

to my family and my beloved

Acknowledgments

This thesis would not have been possible without the support of many people. I would like to express my deepest gratitude to my thesis advisor Ş. İlker Birbil who was abundantly helpful and offered invaluable support and guidance. I am also thankful for his encouraging attitude throughout my undergraduate and graduate studies at Sabancı University. I would also like to thank Kerem Bülbül for his caring and support. I am very grateful to my thesis committee members, Özgür Gürbüz, Güvenç Şahin and Tonguç Ünlüyurt.

I would like to thank TÜBİTAK BİDEB for their financial support during my graduate studies.

I want to offer my special regards to İbrahim Muter for his invaluable assistance and friendship. Without his helpful attitude, it would have become really difficult for me to complete this thesis. My special thanks goes to all my lab friends who made it a convivial place to work. Especially, I would like to thank all members of NC2. I am also really grateful to my dearest friend Selda Tatlıdede who has always been there to support me.

I owe my loving thanks to Oğuzhan Uçar who is the origin of my happiness. His encouragement and support carried me through the roughest times.

Last but not the least, I want to express my eternal gratitude and blessings to my family for their concern and love throughout my life. Besides this thesis, without their endless support, I would not be who I am.

MULTI-DEPOT VEHICLE SCHEDULING WITH DISRUPTIONS

Ezgi Yıldız

Industrial Engineering, Master of Science Thesis, 2011

Thesis Supervisor: Assoc. Prof. Ş. İlker Birbil

Keywords: multi-depot vehicle scheduling, disruption management, column generation, column and row generation

Abstract

This thesis aims at solving a multi-depot vehicle scheduling problem with disruptions at the operational level. Although there are many cases of disruptions in transportation, we mainly focus on the excessive customer demand for a specific destination and late arrival of vehicles. We refer to these problems as extra trip and delayed trip problems, respectively. Then we propose a solution method that is based on swapping two routes. Conventional vehicle scheduling problem is solved by generating a sequence of trips known as routes. By using a set partitioning model, it is ensured that each trip is covered exactly once while the total deadheading cost is minimized. Since the set partitioning model has exponentially many variables, the column generation algorithm is used to solve the problem efficiently. However, with our proposed swapping strategy, a set partitioning model with side constraints is formulated. To solve this new model, we need to use two different subproblems. The first one is a standard one, which follows the column generation algorithm directly. The second subproblem generates pairs of routes (columns) and along with these pairs constraints are added to the problem. To handle this difficulty, we propose to apply a column-and-row generation algorithm and discuss the optimality of this approach. The proposed method is tested on a set of randomly generated problem instances. Computational results show that the proposed approach can effectively handle the disruptions at the expense of a slight increase in the cost of the conventional model.

ÇOK DEPOLU ARAÇ ROTALAMA PROBLEMLERİNDE AKSAKLIKLAR

Ezgi Yıldız

Endüstri Mühendisliği, Yüksek Lisans Tezi, 2011

Tez Danışmanı: Doç. Dr. Ş. İlker Birbil

Anahtar Kelimeler:çok depolu araç rotalama problemi, aksaklık yönetimi, sütün türetme, satır ve sütün türetme

Özet

Bu çalışma, çok depolu araç rotalama probleminde operasyon aşamasında ortaya çıkabilecek aksaklıkları yönetmeyi amaçlamaktadır. Ulaşım sektöründe, ilgili problemler yapısı itibari ile çok farklı aksaklıklar içerse de, bu tez kapsamında özellikle aşırı müşteri talebi ve gecikmiş seferler göz önünde bulundurulmuştur. Sözü edilen aksaklıklar ek sefer ve gecikmiş sefer problemleri olarak isimlendirilmiştir. Bu iki probleme çözüm olarak, uygun iki rotayı çaprazlayan bir yöntem geliştirilmiştir. Geleneksel araç rotalama problemi her bir seferi bir kez kapsayacak şekilde seferler dizisinden oluşan rotaların yaratılması ve bu rotaların küme bölüntüleme modelinde kullanılması ile çözülür. Amaç fonksiyonunun, toplam boş sefer mesafesini en küçüklemek olduğu bu model üssel sayıda değişken içerdiğinden sütun türetme yaklaşımı kullanılır. Fakat aksaklıkların yönetimi için küme bölüntüleme modeline eklenen yan kısıtlar modelin standart sütun türetme yöntemi ile çözümünü imkansız hale getirmiştir. Bu yeni modeli çözebilmek için iki farklı alt probleme ihtiyaç duyulmuştur. Alt problemlerden ilki tek bir rota oluşturulmasını amaçlarken diğer problem rota çifti oluşturmaktadır. Rota çiftleri oluşturan bu problem değişkenler ile birlikte kısıtlarıda beraberinde getirdiği için satır-ve-sütun türetme yaklaşımı geliştirilmiş ve yöntemin en iyilik durumu ispat edilmiştir. Önerilen yöntemin hesaplama sonuçlarının gösterdiği üzere elde edilen sonuçlar aksaklıklara çözüm sunarken, geleneksel yöntemin amaç fonksiyonunda sınırlı miktarda artışa sebep olmuştur.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

The planning process of transportation includes different phases. It starts with the strategic level decision of demand forecasting and ends with a set of lines that will be opened. As the next step, the set of trips are determined with specific departure and arrival destinations. Then each trip obtains a departure and arrival time. After this setup, it is necessary to assign the vehicles to the scheduled trips, and the resulting problem is called as vehicle scheduling problem (VSP).

VSP is an important stage of transportation planning. Since this problem is solved frequently by the transportation companies, many researchers are interested in solving VSP. The major works in literature focus on two problems; single depot vehicle scheduling problem (SDVSP) and the multi-depot vehicle scheduling problem (MDVSP). To solve SDVSP, polynomial time algorithms, such as quasi-assignment proposed by Freling et al. [2001] are available. However, when there are multiple depots, the problem becomes NP hard (see Daduna et al. [1993] for details). One of the well-known approaches to solve MDVSP is based on generating a sequence of trips that will be assigned to a vehicle.

In this thesis, we focus on MDVSP for inter-city transportation. To solve this problem, we first employ a network structure. In the network model, each trip is represented by a node and the compatibility relations are represented by arcs. For each depot, a source and a sink node are introduced to the network. There is a connection from each depot to a trip and likewise, from each trip to each depot. An arc cost is equal to the distance between the arrival location of the head node and the departure location of the tail node of the arc. These costs are related to relocation costs, which are also called the deadheading costs especially in the airline scheduling literature.

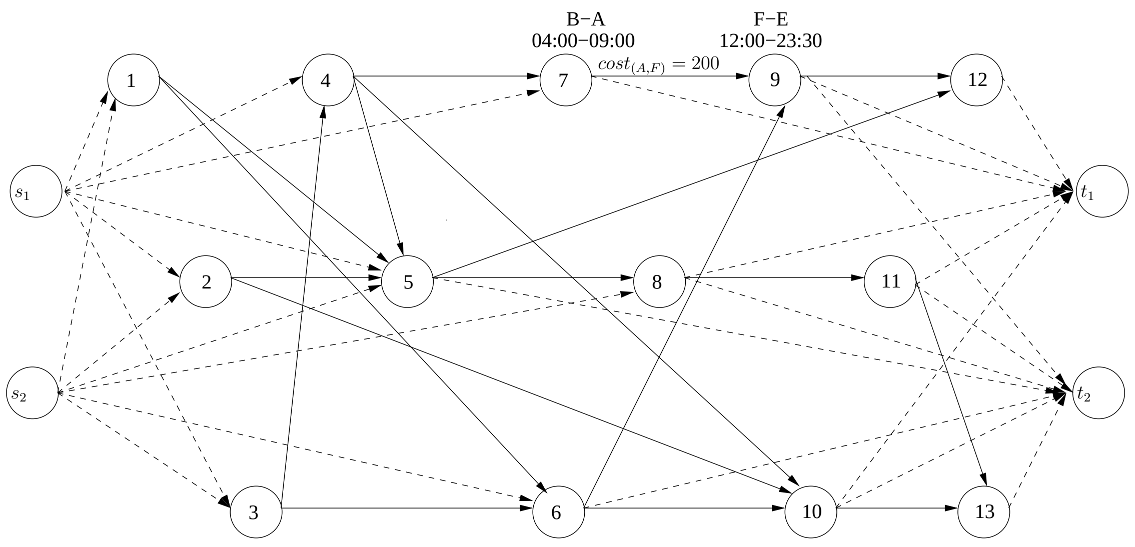


Figure 1.1: Sample network

In Figure 1, a sample network is illustrated. Two dummy source nodes, s_1 and s_2 and two dummy sink nodes, t_1 and t_2 are inserted for two depots, respectively. In a given sequence, a pair of consecutive trips are said to be compatible, if the time windows of these trips do not overlap and there is sufficient time (connection time) between them. In sample network, $(7, 9)$ is a compatible pair since a vehicle can complete trip 7 at 09:30 and travel from city A to city F, which takes 4 hours, and then take over trip 9. The cost of arc $(7, 9)$ is the travel distance between city A to city F. If the arrival city of a trip is the same as the departure city of the succeeding trip, then the deadheading cost is zero. As emphasized above, to solve the MDVSP, we need to generate sequences of trips, which contain compatible pairs. In the remaining part of this study, this sequence of trips are called as routes . For instance, $s_1 - 2 - 5 - 8 - 11 - t_1$ is a route.

The objective of MDVSP is to select a subset of all feasible routes in order to cover each trip in the schedule exactly once with the minimum total deadheading cost. This problem may be stated as a set partitioning problem with side constraints:

$$\text{minimize} \quad \sum_{r \in \mathcal{R}} c_r y_r, \tag{1.1}$$

$$\text{subject to} \quad \sum_{r \in \mathcal{R}} a_{tr} y_r = 1, \qquad t \in \mathcal{T}, \tag{1.2}$$

$$\sum_{r \in \mathcal{R}} b_r^d y_r \leq C^d, \qquad d \in \mathcal{D}, \tag{1.3}$$

$$y_r \in \{0,1\}, \qquad r \in \mathcal{R}, \tag{1.4}$$

where $\mathcal{R}$ is the set of all routes, $\mathcal{T}$ is the set of all trips and $\mathcal{D}$ is the set of all depots. Here c_r is the cost of route r; $a_{tr} = 1$, if trip t is covered by route r and 0, otherwise. The decision variable y_r equals to 1, if route r is selected and 0, otherwise. Parameter $b_r^d = 1$ if route r leaves depot d and 0, otherwise. C^d is the capacity of depot d. The objective function minimizes the total cost of the selected routes. The set of constraints (1.2) implies that each trip should be covered exactly once. The set of constraints (1.3), ensure that the number of routes assigned to a specified depot is less than the capacity of the depot.

Although MDVSP depends on the set of trips determined by the previous stages of the planing, these decisions sometimes do not overlap with the reality because they are obtained from some forecasts and historical data. Since there are many uncertainties in real life, the schedules may not work and result with extra costs for companies. For instance, because of traffic jam, the arrival time of a vehicle cannot be anticipated correctly or because of the excess demand for a particular destination, an extra trip may be inserted into the schedule. To alleviate the adverse effects of these disruptions, some researchers study mathematical modeling approaches that take into account the possible disruptions at the planning stage. Similarly, in this thesis, we assume that possible sources of disruptions may be guessed, and to handle these disruptions at the operation level, we try to create a robust schedule in the sense that this schedule gives more flexibility to the decision makers to recover from the disruptions.

1.1 Contributions of The Thesis

Main contribution of this thesis is to create a robust solution to MDVSP to manage disruptions. With our proposed solution approach, the delayed and the the extra trips are handled at operational level. We solve the MDVSP with column generation. Although column generation is frequently applied for solving conventional MDVSP, it is not possible to use column generation directly to solve our particular robust model. Thus, we introduce a new subproblem that grows both horizontally (column-wise) and vertically (row-wise). With using two different subproblems, the optimality of the algorithm is proved. The contributions of this study can be listed as follows:

- We give mathematical programming model that takes into consideration the disruptions caused by delays and extra trips in MDVSP.

- We introduce a solution approach to solve the linear programming relaxation of the proposed integer programming model.
- This approach is a special application of the method proposed by Muter et al. [2011].
- We show that the proposed solution approach converges to the optimal solution of the linear programming relaxation of the robust integer programming model.
- We test our method with a variety of randomly generated instances. Our solutions emphasize that:
  - with small cost increases, possible sources of disruptions can be handled
  - large MDVSP instances can be solved in reasonable amount of time.

1.2 Outline

The outline of this thesis is as follows: Chapter 2 gives a literature review on the MDVSP and disruption management. Chapter 3 includes our problem statement and the proposed algorithm with its optimality proof. In Chapter 4, we present a computational study on a set of randomly generated problem instances. Conclusions are discussed in Chapter 5.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we present the existing modeling approaches and solution techniques that are used to formulate and solve the multi-depot vehicle scheduling problem. We also present the studies focused on the disruption management in transportation. Existing solution approaches based on column-and-row generation are also included in the last part of this chapter.

2.1 Multi-Depot Vehicle Scheduling

In the literature, the early solution methods for multi-depot vehicle scheduling problem (MDVSP) are mainly the heuristic methods. To the best of our knowledge, the first studies using heuristics are given by Bodin et al. [1983] and Bertossi et al. [1987].

Dell'Amico et al. [1993] apply a polynomial time heuristic based on solving a series of shortest path problems and minimizing the total number of vehicles. In their study, they try to minimize the total travel distance by covering all trips. To solve the problem, they use a network representation and determine the duty of a vehicle with a circuit on this network. Their solution algorithm includes a series of stages. At the first stage, they solve the relaxation of the original problem, where each cycle can include more than one depot. This relaxed version is a simple problem that can be solved easily.

Haghani and Banihashemi [2002] also apply heuristics to solve an extension of MDVSP problem, namely, multiple depot vehicle scheduling problem with route time constraints (MDVSRTC). For this problem, they add a constraint which limits the duration of a route. As a solution procedure, they use an exact method and two different heuristic approaches. Since the size of the constraint set that limits the route time, is inherently large, it is not possible to solve the problem directly. Instead, they apply a constraint generation method. As a heuristic approach, first they solve the problem without the route time limiting constraint. For the routes that violate the time limit, they delete the trips one by one until the route becomes feasible. They resolve the model and apply the procedure until all routes are feasible. With the second heuristic, they aim to reach integer solution. Furthermore, in their study they try to decrease the total number of joining trips and the decision variables.

Beyond the heuristics, several exact methods are also applied to solve MDVSP. Ribeiro and Soumis [1994] formulate MDVSP as a set partitioning problem. In our study, we also adopt the same formulation. Ribeiro and Soumis apply some bounding techniques and relax the MDVSP with different techniques. With these techniques, they come up with assignment and shortest path problems. They also use additive bound technique, which is also used by Carpaneto et al. [1989a]. In this study, Ribeiro and Soumis show that, the linear relaxation of MDVSP generates a better bound than the additive bound of Carpaneto.

In his thesis, Löbel [1998] tries to solve a large scale MDVSP by using combinations of several methods. First, he applies Lagrangian relaxation to set a tight lower bound on the fleet size and the operational cost. For this bound, he uses two different Lagrangian relaxation. Then, he searches for a feasible integer solution as a good starting point. By using two different primal opening heuristics. The first heuristic assigns each trip to one of the depots with the shortest travel distance. The second heuristic is based on the minimum-cost flow idea. After determining the initial feasible solution, linear programming (LP) relaxation of the problem is solved by column generation. To test the algorithm, he uses a real-world data involving the cities of Germany.

Similar to Ribeiro and Soumis, Hadjar et al. [2006] apply a column generation method to solve MDVSP. Since the LP relaxation optimal solution may include some fractional variables, they introduce a class of valid inequalities to obtain integer solutions. By using the newly introduced constraints, they create a new reduced cost formulation to solve the resulting problem. Since the number of variables of MDVSP is huge, they also use variable fixing procedure. In their study, they define the non-integral solution of the linear relaxation of the problem as an odd cycle. To remove the odd cycles they apply lifting procedures to MDVSP . They also use a branch-and-bound tree and apply column generation at each node. This approach is also known as branch-and-price.

2.2 Managing Disruptions

In the literature, the solution methods to manage disruptions can be categorized under two groups. The first group focuses on the online approach. These studies compute the schedules of vehicles during the execution and update the schedules at the operational level. Alternate solution approaches consider the possible disruptions at the planning stage. This class of solution procedures creates schedules that are likely to remain stable when disruptions occur. These approaches belong to the second group and they are called offline aproaches. The proposed approach in this thesis is also an offline approach.

As an example of online solution approach, Huisman et al. [2004] underline the importance of the delays in the schedules of public transportation vehicles. First, they apply a static model using multi-commodity flow formulation. Then, they focus on a dynamic solution approach which just sets the schedule of the near future. They assume that the travel times in that period are known. For the following periods, the probabilities of occurrences are known and these probabilities are based on historical data, subjective expert opinions, or a combination of both. In fact, they solve the problem at time T, where the schedule for $[T, T + l]$ is created with using the probabilistic information of the period after $T + l$. According to the computational results, by increasing the number of vehicles they can reduce the number of delayed trips with the dynamic method.

Li et al. [2008] focus on the breakdowns of vehicles as a source of disruption. They state that, it is necessary to pick up the passengers of a broken vehicle and then complete the remaining trips of the vehicle to avoid any delays and cancelations. With their proposed method, they consider the operation costs, the scheduled disruption costs and the trip cancelation costs for the single depot vehicle scheduling problem. Their solution method starts with preprocessing step, which generates the network. Then, they apply Lagrangian relaxation with relaxing one of the constraints of the mathematical model. They also decompose the relaxed problem into three subproblems. To solve these subproblems, they apply column generation. With the Lagrangian multipliers, the network is updated and the shortest path problems on reduced network is solved. Their results show that the parameters like trip cancelation costs or penalties on each reassignment clearly affect the optimal solution of the problem. In this thesis, we also apply column generation, however, instead of Lagrangian multipliers, we use the dual values. With Lagrangian heuristic method, Li et al. solve at most 700 original trip problem. However, as illustrated in section 4, we can solve 800 trip problem with 2 depots.

In the study of Kramkowski et al. [2009], the possible delays that can occur at the operational level are handled by inserting buffer times between the trips. The critical point of this approach is to determine the right place to insert the buffer times. To solve the resulting robust vehicle scheduling problem, they apply a path based flow decomposition method on a network. For different objectives, different decomposition strategies are used. As another method, they manually add idle times before and after each trip. They apply simulated annealing to solve the problem. In this setting, they represent the objective function of the model in terms of the planned and the delayed cost. Since the delay amounts are not known certainly, the objective function are stochastically influenced. Thus, instead of conventional simulated annealing, they apply an extension called simulated annealing for noisy environment algorithm. When these different approaches are compared in terms of their solution performance the decomposition strategies and simulated annealing extension is shown to perform best.

Sato et al. [2009] develop a method to manage disruptions that can be applied both on vehicle and train scheduling. Mainly, their study focuses on the network-flow model and compare it with the set-covering formulation. As a solution method to manage disruptions, they develop a two phase heuristic. At the first phase, they partially swap the schedule of vehicles or trains to generate a feasible solution by modifying the original schedule. The second phase is a local search heuristic which tries to improve the solution while maintaining feasibility. It is clear that the idea behind to manage disruptions is similar to our the solution approach. However, our approach yields an optimal solution.

Managing disruption is also studied in the airline scheduling literature. Shebalov and Klabjan [2006] attack robust crew scheduling problem by paying attention to the delays of flights. With their proposed method, they try to include as many recovery combinations as possible within the optimal solution. As a recovery method for each flight, the keep a different crew to cover the preceding flights of this trip and call this crew as the move-up crew. With this method, the duties of two crews are swapped, if a delay occurs for the specified flight. This idea to create the recovery solution is also used in our study. Their model is based on standard set partitioning problem. By adding a side constraint, they also keep the operation cost less than a threshold value. To solve the overall mathematical model, they use a combination of Lagrangian relaxation and column generation.

In their study Tekiner et al. [2009] focus on a specific disruption in airline crew pairing disruption is caused by the extra flights that could be added to the flight schedule at the operational level. To handle this disruption at the planning stage, they come up with two solution approaches. One of the approaches try to generate pairings that can directly cover extra flight. Other approach focuses on swapping duties of two crews. To model the problem, they use a set covering formulation with side constraints and solve the resulting problem by applying a column generation based heuristic.

2.3 Column-and-Row Generation

Column generation is a well-known approach for solving large-scale LP problems. It is commonly used to solve problems with exponentially many variables. However, with conventional column generation, it is not possible to solve problems that expand both horizontally (column-wise) and vertically (row-wise). Thus some researchers propose solution method that generate both columns and rows.

Muter et al. [2011] propose a simultaneous column-and-row generation algorithm. Their study focuses on a set of of large-scale LP problems with exponentially many variables and column dependent rows. In other words, these problems have a common issue that the generation of variables (columns) creates sets of linking constraints. To solve these problems, they propose a simultaneous column-and-row generation algorithm with three subproblems. They also discuss the optimality condition of this solution method and illustrate its application to the multi-stage cutting stock and the quadratic set covering problems. In this thesis, we apply the column-and-row generation approach developed by Muter et al.. The proposed method of Muter et al. has a generic form for each horizontally and vertically expanding problem, we develop a specific version of it and prove its optimality.

Another study that discusses simultaneous column and row generation is proposed by Feillet et al. [2010]. In this study they develop a method that combines branch-and-cut with branch-and-price. Since their methodology includes simultaneous generation of variables and cuts, their problems expand both horizontally and vertically. For a restricted master problem, they construct a primal feasible but not necessarily a dual feasible solution such that cost values of the both of the problems equal. If the existing solution is dual feasible, they terminate with an optimal solution. Otherwise, at least one variable, which is associated with the violated dual constraint is added to the restricted master problem. In their study, they illustrate implementations for two problems; split delivery vehicle routing problem and a service network design problem for urban rapid transit systems.

CHAPTER 3

MULTI-DEPOT VEHICLE SCHEDULING WITH DISRUPTIONS

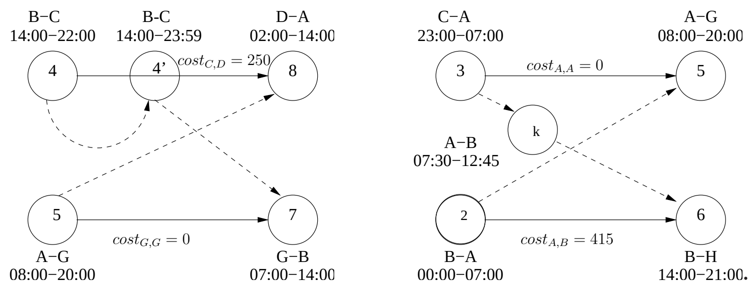
In this chapter, we give the mathematical model of our multi-depot vehicle scheduling problem with disruptions. We explain our solution approach prove the optimality of the proposed approach.

3.1 Problem Statement

We focus on the disruptions that may occur at the operational level. In the scope of this thesis, we attack two types of disruptions; delays and extra trips. The delayed trips problem aims to cover the predetermined trips with the existing vehicles in case of possible delays. Similar to the delayed trips problem, the extra trips problem deals with possible additional trips that can be added to the schedule at the operational level. These extra trips must be covered by adjusting the planned schedule with the minimum number of changes. Although these two problems are seemingly different, we may apply the same approach to solve them. This approach is based on swapping two routes in such a way that if any one of these disruptions occurs, then the recovery is possible with the existing route schedule.

If a trip is delayed and it may cause further delays in all subsequent trips. Hence all those subsequent trips that are delayed can be continued with another vehicle that has ample time to take over. Likewise, the delayed vehicle shall then carry on with the subsequent trips of the latter vehicle. This is called swapping routes. Similar to delayed trips, extra trips can also be handled by swapping routes. One of the vehicles attempts to cover the extra trip, and hence cannot continue with the remaining trips in its route. A second vehicle then swaps its remaining trips with those of the first vehicle so that the remaining trips of the first vehicle are also covered. As it is common in practice, the exact time of the extra trips are not known in advance, and hence, we assume that a time window is given for the possible departure time of the extra trips.

We take the worst case scenario and keep the earliest departure and latest arrival time for each extra trip.



(a) Delayed Trip (b) Extra Trip

Figure 3.1: Swapping solutions for disruptions

The swapping operation for delays is illustrated in Figure 3.1(a). B is the departure, C is the arrival city for trip 4. The departure and arrival times are also given. Since the distance from city C to city D is 250 kilometers, the cost of the arc between node 4 and node 8 is 250 units. In the original assignment, one of the vehicles covers the trip 4 and continues with trip 8, and another vehicle covers 5 and 7 in this order. However, trip 4 is delayed and the vehicle reaches city C two hours later. Consequently, it is not possible for this vehicle to reach city D in two hours because travel time between city C and D is only three hours. Instead of delaying trip 8 as well, the company has an opportunity to recover this disruption with another vehicle. That is, if the disruption from this delay occurs, the vehicles who cover the trips 4, 8 and 5, 7 swap their duties. New sequence will be $4'$, 7 and 5, 8. In Figure 3.1(b), without the extra trip k one of the vehicles covers trips 3 and 5, and another vehicle covers trips 2 and 6. If the extra trip k is realized, the first vehicle covers trip 3, the extra trip and then continues with trip 6. Other vehicle does trip 2 and continues with trip 5. As illustrated in Figure 3.1, both the delayed trips and the extra trips can be solved by the same method. Only difference between these two problems is the search method to find possible swapping solutions.

By taking into consideration the recovery options for disruptions, we adapt model (1.1)-(1.4) and obtain

$$\text{minimize} \quad \sum_{r\in\mathcal{R}} c_r y_r, \tag{3.1}$$

$$\text{subject to} \quad \sum_{r\in\mathcal{R}} a_{tr} y_r = 1, \qquad t \in \mathcal{T}, \tag{3.2}$$

$$\sum_{r \in \mathcal{R}} b_r^d y_r \leq C^d, \qquad d \in \mathcal{D}, \tag{3.3}$$

$$\sum_{(r,q) \in \mathcal{P}(k)} x_{(r,q)}^k \geq 1, \qquad k \in \mathcal{K}, \tag{3.4}$$

$$y_r \geq x_{(r,q)}^k, \qquad r \in \mathcal{R} : (r,q) \in \mathcal{P}(k), k \in \mathcal{K}, \tag{3.5}$$

$$y_q \geq x_{(r,q)}^k, \qquad q \in \mathcal{R} : (r,q) \in \mathcal{P}(k), k \in \mathcal{K}, \tag{3.6}$$

$$y_r + y_q \leq 1 + x_{(r,q)}^k, \qquad (r,q) \in \mathcal{P}(k), k \in \mathcal{K}, \tag{3.7}$$

$$y_r \in \{0,1\}, \qquad r \in \mathcal{R}, \tag{3.8}$$

$$x_{(r,q)}^k \in \{0,1\}, \qquad (r,q) \in \mathcal{P}(k), k \in \mathcal{K}, \tag{3.9}$$

where $\mathcal{K}$ denotes the set of all possible disruptions (delayed trips or extra trips insertions). The index set $\mathcal{P}(k)$ denotes the set of route pairs that can be swapped to create a solution for disruption k. The auxiliary binary variable $x_{(r,q)}^k$ is set to 1, if the route pair $(r,q) \in \mathcal{P}(k)$ and both routes r and q are presented in the route solution and to 0, otherwise. The sets of constraints (3.5)-(3.7) prescribe that $x_{(r,q)}^k$ takes the value 1 if and only if $y_p = y_q = 1$. They are called the linking constraints. Notice that the number of linking constraints depends on the total number of swapping solutions. We name this new model as MDVSP with disruptions.

3.2 Column-and-Row Generation Approach

In almost all practical settings, the total number of routes (variables) in model (3.1)-(3.9) is very large. Therefore, it is common to use column generation to solve the problems with large number of variables (see Dantzig and Wolfe [1960]). The column generation algorithm aims to solve the LP relaxation of model (3.1)-(3.9). The linear programming model becomes

$$\text{minimize} \quad (3.1), \tag{3.10}$$

$$\text{subject to} \quad (3.2) - (3.7), \tag{3.11}$$

$$0 \leq y_r \leq 1, \qquad r \in \mathcal{R}, \tag{3.12}$$

$$0 \leq x_{(r,q)}^k \leq 1, \qquad (r,q) \in \mathcal{P}(k), k \in \mathcal{K}. \tag{3.13}$$

In this study, we call this model (3.10)-(3.13) as the master problem (MP). In traditional column generation to solve MDVSP, the restricted master problem (RMP) contains all the constraints but only a subset of variables. Using the optimal dual values of RMP, a pricing subproblem is solved, where a new column with a negative reduced cost is searched. The pricing subproblem for MDVSP is typically solved on a trip network (see Figure 1.1). The objective of this subproblem is to find a path from a specified depot to the same depot with the minimum reduced cost. In this thesis, we shall call this standard subproblem as the individual route pricing subproblem.

In column generation, all constraints of the model are assumed to be available. However, constraints (3.5)-(3.7) depend on the decision variables through the set $\mathcal{P}(k)$. Thus, it is not possible to solve problem (3.1)-(3.9) with conventional column generation approach. Because of these column-dependent-rows, the problem has to be initialized with a subset of rows and columns. Therefore, following its definition in Muter et al. [2011], the resulting problem is called as the short restricted master problem (SRMP), since we not only have a subset of columns but also a subset of rows. To solve the LP relaxation of MDVSP by column generation, we replace the sets $\mathcal{R}$ and $\mathcal{P}(k)$ by their subsets, namely $\overline{\mathcal{R}}$ and $\overline{\mathcal{P}}(k)$, respectively. The resulting model is then given by

$$\begin{aligned}
\text{minimize} \quad & \sum_{r\in\overline{\mathcal{R}}} c_r y_r, && && (3.14)\\
\text{subject to} \quad & \sum_{r\in\overline{\mathcal{R}}} a_{tr} y_r = 1, && t \in \mathcal{T}, && (3.15)\\
& \sum_{r\in\overline{\mathcal{R}}} b_r^d y_r \le C^d, && d \in \mathcal{D}, && (3.16)\\
& \sum_{(r,q)\in\overline{\mathcal{P}}(k)} x_{(r,q)}^k \ge 1, && k \in \mathcal{K}, && (3.17)\\
& y_r \ge x_{(r,q)}^k, && r \in \overline{\mathcal{R}} : (r,q) \in \overline{\mathcal{P}}(k), k \in \mathcal{K}, && (3.18)\\
& y_q \ge x_{(r,q)}^k, && q \in \overline{\mathcal{R}} : (r,q) \in \overline{\mathcal{P}}(k), k \in \mathcal{K}, && (3.19)\\
& y_r + y_q \le 1 + x_{(r,q)}^k, && (r,q) \in \overline{\mathcal{P}}(k), k \in \mathcal{K}, && (3.20)\\
& 0 \le y_r \le 1, && r \in \overline{\mathcal{R}}, && (3.21)\\
& 0 \le x_{(r,q)}^k \le 1, && (r,q) \in \overline{\mathcal{P}}(k), k \in \mathcal{K}. && (3.22)
\end{aligned}$$

According to the model above, when (r,q) in $\mathcal{P}(k)\backslash\overline{\mathcal{P}}(k)$ is added to the model, a new set of constraints of type (3.18)-(3.20) and a variable $x_{(r,q)}^k$ are also introduced to the model. If any one of y_r and y_q is not part of the SRMP, it can be emphasized that the linking constraints are redundant.

A typical column generation algorithm creates a variable, adds it to the RMP and solves it to update the values of the dual variables. However, to solve problems (3.1)-(3.9), we also need to find pairs of variables, which triggers the generation of auxiliary x-variables and the associated linking constraints. The proposed column generation algorithm with two pricing subproblems is illustrated in Figure 3.2. This approach is in fact inspired by the column-and-row generation algorithm proposed by Muter et al. [2011] because the mathematical model (3.14)-(3.22) is a special case of the model in that study. At the initialization step of the algorithm, a set of routes is generated to obtain an initial feasible solution for SRMP. After solving the SRMP, the optimal values of the dual variables are obtained. Then, individual route pricing subproblem is solved. If the minimum reduced cost is negative, then the route is added to SRMP and the algorithm continues; otherwise the first phase of the algorithm is terminated. At this phase, a group of columns have already been generated and some of them may become candidates for swapping to create a recovery solution. Therefore, the column pool is searched for possible pairs and for each pair, the auxiliary x variable and the associated linking constraints are added to SRMP. This step is followed by the second phase of the algorithm, where pairs of routes are generated in the second pricing subproblem. For each disruption, we check whether there is any negative reduced cost column only after pairs of routes are generated along with a set of linking constraints. If so, the columns and rows are added to SRMP and the algorithm returns back to the first pricing problem. If the algorithm cannot find such pairs of routes for all disruptions, then the current solution becomes optimal. To emphasize that our algorithm is an application of CRG given in Muter et al. [2010], we remark that individual route generating subproblem corresponds to the y- generating subproblem and route-pair generating subproblem corresponds to the row-generating subproblem in their work.

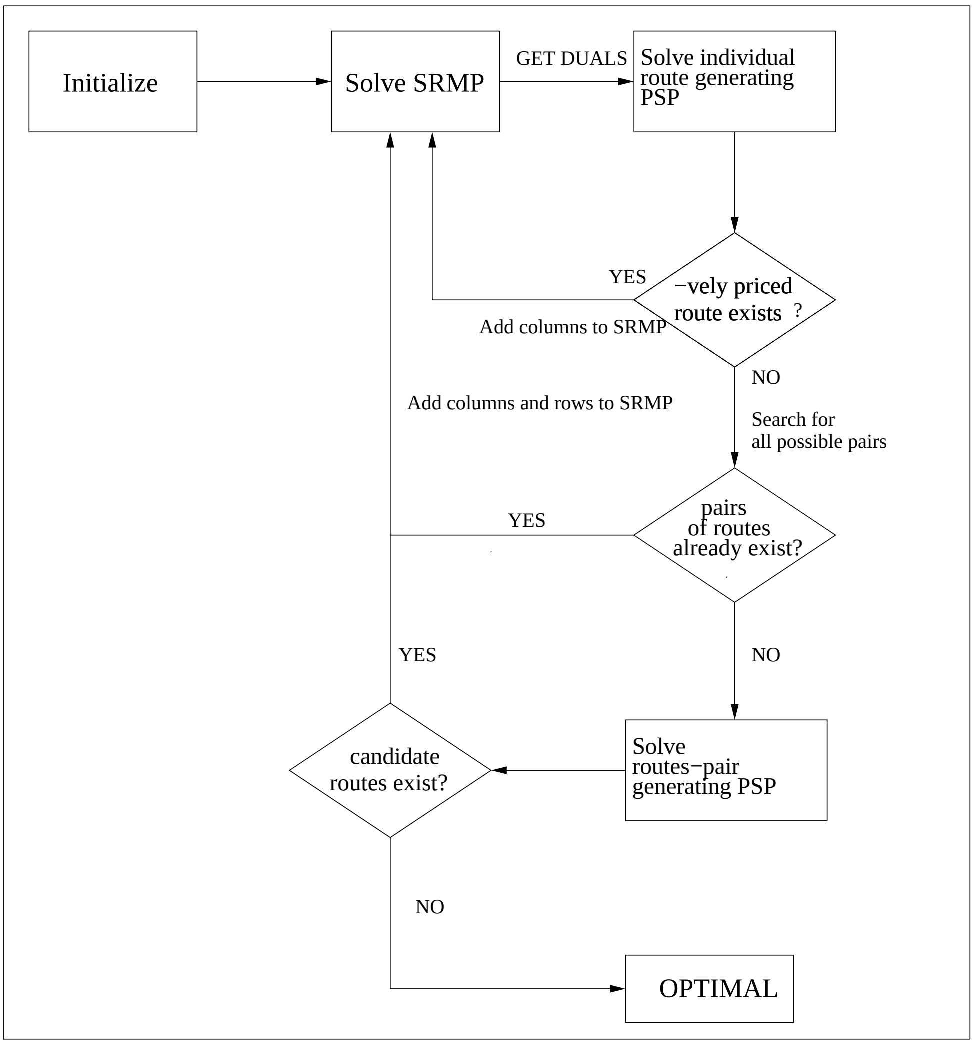

Initialize
Solve SRMP
GET DUALS
Solve individual route generating PSP
YES
−vely priced route exists ?
Add columns to SRMP
NO
Add columns and rows to SRMP
Search for all possible pairs
YES
pairs of routes already exist?
YES
NO
candidate routes exist?
Solve routes−pair generating PSP
NO
OPTIMAL

Figure 3.2: Flow of the proposed column-and-row generation algorithm

For pricing subproblems, the reduced cost evaluation is the key element. The reduced cost of a column is simply checking the violation in the associated dual constraints. To this end, we need the dual of the SRMP. If we denote the dual variables corresponding to the constraints (3.14)-(3.22) by u_t, v^d, z_k, $\gamma^1_{(r,q),k}$, $\gamma^2_{(r,q),k}$, $\gamma^3_{(r,q),k}$,δ^1_r and $\delta^2_{(r,q),k}$ respectively, then the dual problem becomes

$$\begin{aligned}
\text{minimize} \quad & \sum_{t\in\mathcal{T}} u_t + \sum_{d\in\mathcal{D}} C^d v^d + \sum_{k\in\mathcal{K}} z_k + \sum_{k\in\mathcal{K}} \sum_{(r,q)\in\overline{\mathcal{P}}(k)} \gamma^1_{(r,q),k} + \\
& \sum_{r\in\overline{\mathcal{R}}} \delta^1_r + \sum_{k\in\mathcal{K}} \sum_{(r,q)\in\overline{\mathcal{P}}(k)} \delta^2_{(r,q),k}, \qquad (3.23) \\
\text{subject to} \quad & \sum_{t\in\mathcal{T}} a_{tr} u_t + \sum_{d\in\mathcal{D}} b^d_r v^d + \\
& \sum_{k\in\mathcal{K}} \sum_{(r,q)\in\overline{\mathcal{P}}(k)} (\gamma^1_{(r,q),k} + \gamma^3_{(r,q),k}) +
\end{aligned}$$

$$\sum_{k\in\mathcal{K}}\sum_{(q,r),k\in\overline{\mathcal{P}}(k)}(\gamma^2_{(q,r),k}+\gamma^3_{(q,r)})+\delta^1_r\leq c_r,\quad r\in\overline{\mathcal{R}}, \tag{3.24}$$

$$z_k-\gamma^1_{(r,q),k}-\gamma^2_{(r,q),k}-\gamma^3_{(r,q),k}+\delta^2_{(r,q),k}\leq 0,\quad (r,q)\in\overline{\mathcal{P}}(k),k\in\mathcal{K} \tag{3.25}$$

$$v^d\leq 0,\quad d\in\mathcal{D}, \tag{3.26}$$

$$z_k\geq 0,\quad k\in\mathcal{K}, \tag{3.27}$$

$$\gamma^1_{(r,q),k}\geq 0,\quad (r,q)\in\overline{\mathcal{P}}(k),k\in\mathcal{K}, \tag{3.28}$$

$$\gamma^2_{(r,q),k}\geq 0,\quad (r,q)\in\overline{\mathcal{P}}(k),k\in\mathcal{K}, \tag{3.29}$$

$$\gamma^3_{(p,q),k}\leq 0,\quad (p,q)\in\overline{\mathcal{P}}(k),k\in\mathcal{K}, \tag{3.30}$$

$$\delta^1_r\leq 0,\quad r\in\overline{\mathcal{R}}, \tag{3.31}$$

$$\delta^2_{(r,q),k}\leq 0,\quad (r,q)\in\overline{\mathcal{P}}(k),k\in\mathcal{K}. \tag{3.32}$$

We need to maintain primal feasibility throughout column and row generation so that the optimality is reached by the eliminating columns with negative reduced costs. At any iteration the missing constraints (3.15)-(3.17) that are not in SRMP are not violated because, unless both y_r and y_q are generated, the associated linking constraints are redundant. Moreover, if only one of y_r or y_q exists in the model, then keeping the auxiliary variable $x^k_{(r,q)}$ as nonbasic and the bounds on the constraints ensures that the corresponding constraints are not violated. Overall the primal feasibility is always preserved as in traditional column generation. Thus, we next focus on the dual feasibility by solving the pricing subproblems.

3.2.1 Individual Route Generating Subproblem

According to the proposed algorithm, after the SRMP is solved with the existing variables and constraints, the individual route pricing subproblem is called to check if there is any route with a negative reduced cost. Using the optimal values of the known dual variables obtained by solving SRMP, we check whether any of the dual constraints (3.24)-(3.25) is violated. If we denote the reduced cost for route $r\in\overline{\mathcal{R}}$ is by $\overline{c}_r$, then we have

$$\overline{c}_r=c_r-\sum_{t\in\mathcal{T}}a_{tr}u_t-\sum_{d\in\mathcal{D}}b^d_r v^d-\sum_{k\in\mathcal{K}}\sum_{(r,q)\in\overline{\mathcal{P}}(k)}(\gamma^1_{(r,q),k}+\gamma^3_{(r,q),k})-$$

$$\sum_{k \in \mathcal{K}} \sum_{(q,r),k \in \overline{\mathcal{P}}(k)} (\gamma^2_{(q,r),k} + \gamma^3_{(q,r)}) - \delta^1_r. \tag{3.33}$$

Since route r has not been generated yet, it cannot appear in any swapping solution set in $\overline{\mathcal{P}}(k)$. Moreover, since we produce only an individual route, we are not interested in whether this route pairs up with any other individual route. Hence, in relation (3.33), the variables $\gamma^1_{(r,q),k}$, $\gamma^2_{(r,q),k}$ and $\gamma^3_{(r,q),k}$ are equal to zero for $(r,q) \in \overline{\mathcal{P}}(k)$. Also δ^1_r is equal to zero because $y_r = 0$. This leads to

$$\overline{c}_r = c_r - \sum_{t \in \mathcal{T}} a_{tr} u_t - \sum_{d \in \mathcal{D}} b^d_r v^d. \tag{3.34}$$

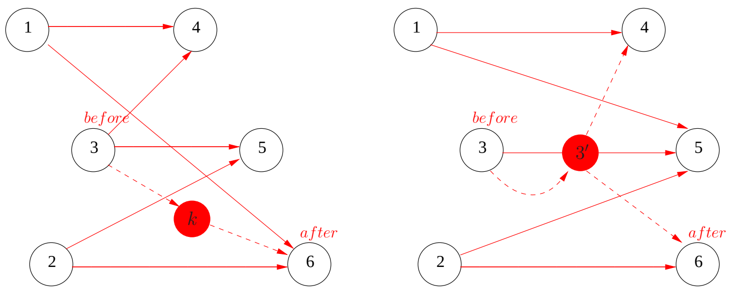
The objective of the pricing subproblem is to determine a route r with $\overline{c}_r < 0$. Solving the pricing sub-problem of the VSP is equivalent to solving the shortest path problem on the network representation where the weights correspond to the dual values (see also Desrochers and Soumis [1989]).

We should emphasize that SRMP enlarges column-wise but it is fixed row-wise until route pricing subproblem cannot find negatively priced column. During individual route pricing, some column pairs that can be swapped to handle the disruption may be incidentally formed. However, their associated linking constraints (3.18)-(3.20) and the auxiliary variables are not a part of SRMP. Before starting the column pairs pricing subproblem, we investigate the pool of columns to determine the possible swapping solutions and add necessary constraints along with the auxiliary variables. After adding these rows coming from the existing columns, we solve the resulting SRMP and move on to the route-pair generating subproblem.

3.2.2 Route-Pair Generating Subproblem

By solving this subproblem, our objective is to identify new columns that price out favorably only after adding new linking constraints currently absent from SRMP. To generate new linking constraints, we have to generate route pairs that are in $\mathcal{P}(k) \setminus \overline{\mathcal{P}}(k)$. To find out such pairs of routes, we use a search mechanism based on labeling the nodes for each disruption. We remark that this labeling is done only once throughout the solution procedure.

The search mechanism is illustrated in Figure 3.3. Suppose that the cause of disruption is the addition of the extra trip k. We first check through all trips list and label them with *before* tags if it is possible to add an arc from this trip to the extra.

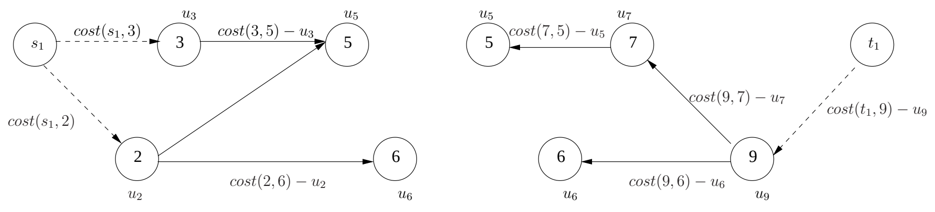


(a) Extra trip (b) Delayed trip

Figure 3.3: Search algorithm for swapping solution of disruptions

Likewise, the trips that can come after the extra trip is labeled by *after* tags. Then for each trip that is labeled as *before*, we check the successor arcs and for each trip that is labeled as *after*, we look at the predecessor arcs. If there is an arc from a predecessor of an *after* trip to a successor of a *before* trip, then these four nodes are candidates for forming a swapping solution for extra trip k when they appear in the routes. In Figure 3.3, trips 3 and 6 are labeled *before*, *after* trips respectively. Trip 5 is the successor to trip 3. On the other hand trips 1 and 2 are the predecessor trips of trip 6. Thus, we should check if there is an arc between 1 to 4, 1 to 5, 2 to 4 or 2 to 5. Since there is an arc from 2 to 5, $(3, 5, 2, 6)$ is a solution for extra trip k. A route containing the connection from node 3 to node 5, and another route containing connection from node 2 to node 6 constitute a candidate recovery pair for extra trip k. In the same vein, $(3, 4, 1, 6)$ is a also solution for the same extra trip. As emphasized before, handling the disruptions caused by the delays are similar to recovering the disruptions from the extra trips. To create a swapping solution for delays, we again try to find a quadruple. However, in this case the first element of quadruple is the delayed trip itself. In Figure 3.3.b, the search mechanism for delayed trip problem is illustrated. Trip 6 is labeled as $after$ because, there is enough connection time between trip 3 after delay and trip 6. Similar to the extra trip problem, $(3, 5, 2, 6)$ and $(3, 4, 1, 6)$ are the sought quadruples.

In the route-pair pricing subproblem, we generate a pair of routes, r and q by solving a shortest path problem. The pricing problem is illustrated in Figure 3.4(a) and 3.4(b).



(a) Original Network (b) Reversed Network

Figure 3.4: Reduced cost calculation for route-pair pricing subproblem

Let $(3,5,2,6)$ be a member of the extra trip k's solution set $\overline{\mathcal{P}}(k)$. To handle the extra trip k, two routes are generated and one of them includes trips 3 and 5, and the other one includes trips 2 and 6. Best way to generate these routes is determining the shortest path from source to nodes 3 and 2, which are already available from the last iteration of individual route generating PSP. To calculate the shortest paths from nodes 5 and 6 to the sink node simply solving the shortest path from each node is not efficient because it results with solving many shortest path problems. Instead, we find the shortest path from the sink to each node on a reversed network, and then solving only one shortest path problem becomes sufficient. Figure 3.4(a) shows the shortest paths to nodes 2 and 3 with the reduced costs on the original network. In Figure 3.4(b), the reversed network is represented with the reversed arcs. This network is used to calculate the shortest paths to nodes 5 and 6. It should be emphasized that the dual value corresponding to constraint (3.16) for depot d should be subtracted from the reduced cost value of route. The same procedure is repeated for the second route. However, we should underline that these two routes must belong to the same depot. This procedure should than be repeated for each disruption.

3.2.3 Optimality of the Proposed Method

In this section, we show the optimality of our methodology outlined in Figure 3.2. We should give the notation that we use in our analysis in Table 3.1.

In the context of a typical column generation approach, equation (3.33) is simplified to equation (3.34). However, when a pair of routes, $(r,q) \in {S_r}^k$ is generated, the associated linking constraints and an auxiliary variable x are also added to SRMP and the dual values of these constraints should also be taken into consideration. To anticipate the dual values of the newly introduced constraints, we assume that $x^k_{(r,q)}$, slack and surplus variables with the associated constraints are added, and the optimal basis is constructed. This optimal basis gives us dual values that are used to calculate

Table 3.1: Notation for analysis

Notation	**Explanation**
$\bar{c}_r$	reduced cost value of y_r induced by B
$\bar{c}_{aug_r}$	reduced cost value of y_r induced by B_r
c_B	objective function values for variables in B
$\bar{y}$	dual values for the constraints in B
$\bar{y}_{aug}$	dual values for the constraints in B_r
A_r	column values of y_r without new constraints
A_{aug_r}	column values of y_r with new constraints
z_k	dual value of constraint set (17) for k^{th} disruption
${S_r}^k = \{(r,q) \in \mathcal{P}(k) \mid \bar{c}_q - z_k < 0\}$	set of variable pairs which satisfy necessary conditions

the reduced cost of the variable pairs. This approach is named as *thinking ahead* approach by Muter et al. [2011]. The rationale of the thinking ahead approach is the following: For a pair (r,q) we first add the associated variable $x^k_{(r,q)}$, the slack and surplus variables and the associated linking constraints set. After this modification this problem is referred to as the augmented problem. The vital step is constructing an optimal basis. Muter et al. propose a way to augment the optimal basis that preserves dual and primal feasibility. Using the resulting values of the dual variables, the reduced costs of the pair of y variables are calculated before actually introducing them to the SRMP. We should emphasize that individual route generating subproblem expands the SRMP in column-wise but do not affect the size of the basis. On the other hand, route-pair generating PSP expands the basis and the SRMP both horizontally and vertically because the sets of linking constraints and their associated variables are added to the SRMP. This expanded problem shall be called as the augmented problem in the remaining part of this thesis. At any stage of the algorithm the basis, denoted by B, has the following form: $B = \begin{pmatrix} A_1 & 0 & E_1 \\ 0 & B_1 & E_2 \\ C_1 & D_1 & E_3 \end{pmatrix}$. The matrix $\begin{pmatrix} A_1 \\ 0 \\ C_1 \end{pmatrix}$ shows the columns of y variables. The matrix $\begin{pmatrix} A_1 & 0 & E_1 \end{pmatrix}$ represents the set of constraints (3.15) and (3.16). The matrix $\begin{pmatrix} 0 \\ B_1 \\ D_1 \end{pmatrix}$ includes the columns of x variables. The matrix $\begin{pmatrix} 0 & B_1 & E_2 \end{pmatrix}$ is associated with constraints (3.17) and, $\begin{pmatrix} E_1 \\ E_2 \\ E_3 \end{pmatrix}$ corresponds to the columns of surplus or slack variables. The matrix $\begin{pmatrix} C_1 & D_1 & E_3 \end{pmatrix}$ represents the existing linking constraints. To construct an optimal basis for the augmented problem, we have to make sure that the reduced cost values of the variables in the augmented problem are greater than or equal to zero. Moreover to calculate the reduced costs correctly, the augmentation must satisfy that the values of the existing dual variables do not change because changes in the dual values affect the reduced cost calculation. We show in Lemma 2 that with the proposed augmentation, which shall be explained later, these conditions are satisfied. Reduced cost for $x^k_{(r,q)}$ variable, denoted by $\overline{d}^k_{(r,q)}$, is given calculated by

$$\overline{d}^k_{(r,q)} = -z_k + \gamma^1_{(r,q),k} + \gamma^2_{(r,q),k} + \gamma^3_{(r,q),k}. \tag{3.35}$$

We have already emphasized that for each route pair, we should add three linking constraint. Thus, we need three new basic variable. Similar to a classical LP sensitivity analysis, initially slack and surplus variables are picked as basic, since we know that the new set of linking constraint currently absent from SRMP are redundant. The dual values of the newly added constraints become zero. Thus, the reduced cost of an $x^k_{(r,q)}$ variable can be calculated as $-z_k$ which is nonpositive. When $z_k = 0$, we can terminate our method. We shall later show in Lemma 3 that this decision is optimal. At this point, we can assume that $-z_k$, is strictly less than zero and according to (3.35) the reduced cost of $x^k_{(r,q)}$ is negative. To preserve dual feasibility, the reduced cost value of x should be greater than equal to zero and according to equation (3.35), the values of $\gamma^1_{(r,q),k}$, $\gamma^2_{(r,q),k}$ or $\gamma^3_{(r,q),k}$ should be equal to z_k. Because of complementary slackness only one of this variable can assume positive value. Since $\gamma^3_{(r,q),k} \leq 0$, the variable x can be basic at newly added rows given by (3.18) or (3.19) and $\gamma^1_{(r,q),k}$ or $\gamma^2_{(r,q),k}$ should have nonzero value. Let $x^k_{(r,q)}$ become basic at constraint (3.18) and $\gamma^1_{(r,q),k} = z_k$, $\gamma^2_{(r,q),k} = 0$ and $\gamma^3_{(r,q),k} = 0$. The augmented basis for only one pair of route, B_r, has the following form

$$B_r = \left(\begin{array}{ccc|ccc} \mathbf{A_1} & \mathbf{0} & \mathbf{E_1} & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{B_1} & \mathbf{E_2} & B_2 & 0 & 0 \\ \mathbf{C_1} & \mathbf{D_1} & \mathbf{E_3} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & -1 & 0 & 0 \\ C_2 & 0 & 0 & -1 & 1 & 0 \\ C_3 & 0 & 0 & -1 & 0 & -1 \end{array}\right). \tag{3.36}$$

Here, the fourth column represents the x variable and the fifth and the sixth columns represent the slack and the surplus variables corresponding to the constraints (3.19) and (3.20), respectively. Note that, at any augmentation, there is a possibility that one of the y_r or y_q variables can be part of the existing basis. If one of the variables is already part of the basis, the associated entries of C_2 and C_3 become 1. We have explained that x variable may pivot out the surplus variables of the rows (3.18) or (3.19). We also emphasize that the ties may be broken arbitrarily. However, to have a valid basis augmentation, the dual values of the existing constraints and the reduced cost values of the existing variables should not change. Let us assume that y_q is already part of the basis and $y_r \in \mathcal{R}\backslash\overline{\mathcal{R}}$. Setting $x^k_{(r,q)}$ as basic for constraint (3.19), is undesirable because in this case the reduced cost of y_q, which is a basic variable, changes and consequently the values of the existing dual variables change. Then, the augmented basis may not be optimal and hence, the reduced costs of y_r is calculated wrongly. Thus, we impose that $x^k_{(r,q)}$ cannot be picked as basic for the constraints which include the existing variables. For this reason, the first three elements of the fourth row are set to zero. The augmented basis notation can now be simplified as

$$B_r = \left(\begin{array}{c|ccc} B & F & 0 & 0 \\ \hline 0 & -1 & 0 & 0 \\ G & -1 & -1 & 0 \\ H & -1 & 0 & 1 \end{array}\right), \tag{3.37}$$

where $F = \begin{pmatrix} 0 \\ B_2 \\ 0 \end{pmatrix}$, $G = \begin{pmatrix} C_2 & 0 & 0 \end{pmatrix}$ and $H = \begin{pmatrix} C_3 & 0 & 0 \end{pmatrix}$.

In the lemmas below, we try to underline the validity of the calculated values of the dual variables and the reduced costs with our specific basis augmentation. It should be emphasized that, instead of only one route pair, for a specific route r, we generate all route pairs (r,q) such that $q \in {S_r}^k$. Thus we use the basis B_r given by

$$B_r = \left(\begin{array}{ccc|ccc} A_1 & 0 & E_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & B_1 & E_2 & \mathbf{B_2} & \mathbf{0} & \mathbf{0} \\ C_1 & D_1 & E_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{C_2} & \mathbf{0} & \mathbf{0} & -\mathbf{I} & -\mathbf{I} & \mathbf{0} \\ \mathbf{C_3} & \mathbf{0} & \mathbf{0} & -\mathbf{I} & \mathbf{0} & \mathbf{I} \end{array}\right), \tag{3.38}$$

where the auxiliary variable $x^k_{(r,q)}$, the slack variables, surplus and the variables for each member of ${S_r}^k$ are added to the basis. For simplicity, the same type of constraints and variables are grouped together. The fourth row corresponds to constraints of type (3.18) for which x variables are basic. The fifth and the sixth rows correspond to constraints of type (3.19) and (3.20), respectively. Using the similar analysis in Muter et al. [2011], we show that indeed the dual values of the existing variables are not altered by our specific augmentation.

Lemma 1 *The inverse of the augmented basis is obtained as*

$$
{B_r}^{-1} = \left(\begin{array}{c|ccc} B^{-1} & B^{-1}F & 0 & 0 \\ \hline 0 & -I & 0 & 0 \\ GB^{-1} & GB^{-1}F+I & -I & 0 \\ HB^{-1} & HB^{-1}F+I & 0 & I \end{array}\right) \tag{3.39}
$$

Proof. Let $J = \begin{pmatrix} B & F \\ 0 & -I \end{pmatrix}$. We know that $J^{-1} = \begin{pmatrix} B^{-1} & B^{-1}F \\ 0 & -I \end{pmatrix}$.$B_r = \begin{pmatrix} J & 0 \\ M & K \end{pmatrix}$ where $M = \begin{pmatrix} G & -I \\ H & -I \end{pmatrix}$ and $K = \begin{pmatrix} -I & 0 \\ 0 & I \end{pmatrix}$ $B_r^{-1} = \begin{pmatrix} J^{-1} & 0 \\ MJ^{-1} & K^{-1} \end{pmatrix}$. Thus

$$
B_r^{-1} = \left(\begin{array}{c|ccc} B^{-1} & B^{-1}F & 0 & 0 \\ \hline 0 & -I & 0 & 0 \\ GB^{-1} & GB^{-1}F+I & -I & 0 \\ HB^{-1} & HB^{-1}F+I & 0 & I \end{array}\right) \tag{3.40}
$$

□

After augmenting the basis and calculating this inverse by (3.39), we can derive the dual values of both the existing constraints and the newly generated constraints. This step is necessary to calculate the reduced costs of the introduced variables. Next, we show that, the values of the dual variables associated with the current basis do not change as we augment B to B_r.

Lemma 2 *The values of the dual variables associated with the existing constraints do not change.*

Proof. Let the objective function coefficients of the basic variables of the augmented basis $c_{Baug} = \begin{pmatrix} c_B & | & c_x & 0 & 0 \end{pmatrix}$, where c_B is the vector of coefficients for the existing variables, c_x is for the newly entered x variable The remaining zero coefficients are given for the surplus variables. Note that c_x is also zero.

$$\overline{y}_{aug} = \left(\begin{array}{c|ccc} c_B & 0 & 0 & 0 \end{array}\right) \left(\begin{array}{c|ccc} B^{-1} & B^{-1}F & 0 & 0 \\ \hline 0 & -I & 0 & 0 \\ GB^{-1} & GB^{-1}F + I & -I & 0 \\ HB^{-1} & HB^{-1}F + I & 0 & I \end{array}\right) = \left(\begin{array}{c|ccc} c_B B^{-1} & c_B B^{-1}F & 0 & 0 \end{array}\right) \tag{3.41}$$

.

Thus $\overline{y}_{aug} = c_B B^{-1}$, which shows the desired result. $\square$

We should also point out that dual values of the newly generated constraints can be anticipated as claimed above with $\gamma^1_{(r,q),k} = z_k$, $\gamma^2_{(r,q),k} = 0$ and $\gamma^3_{(r,q),k} = 0$. For the case ${S_r}^k = \{(r,q)\}$, using the dual values of the constraints, the reduced costs of y_r and y_q are calculated as

$$\overline{c}_{aug_r} = c_r - \overline{y}_{aug} A_{aug_r} = c_r - \begin{pmatrix} c_B B^{-1} & \mid & z_k & 0 & 0 \end{pmatrix} \left(\begin{array}{c} A_r \\ \hline 1 \\ 0 \\ 1 \end{array}\right) = \overline{c}_r - z_k \tag{3.42}$$

and

$$\overline{c}_{aug_q} = c_q - \overline{y}_{aug} A_{aug_q} = c_q - \begin{pmatrix} c_B B^{-1} & \mid & z_k & 0 & 0 \end{pmatrix} \left(\begin{array}{c} A_q \\ \hline 0 \\ 1 \\ 1 \end{array}\right) = \overline{c}_q, \tag{3.43}$$

respectively. From traditional column generation perspective, if one of the variables in any set $(r,q) \in {S_r}^k$ has negative reduced cost, we add the associated y variable to the RMP and then column generation continues. However as we show in the following lemma, we propose a stronger condition to terminate, where we take the thinking-ahead approach a step further.

Lemma 3 *Given an optimal basis B_r and a set of associated optimal values for the dual variables, the proposed algorithm terminates with an optimal solution for the master problem (MP) when*

$$\min_{r \in (\mathcal{R} \setminus \overline{\mathcal{R}})} \{\overline{c}_r + \textstyle\sum_{q \in {S_r}^k} \overline{c}_q - \mid {S_r}^k \mid z_k\} \geq 0.$$

Proof. Before we first assume that $z_k = 0$. In this case ${S_r}^k$ is an empty set and $\{s_1, s_2, s_3\}$ for all $(r,q) \in \mathcal{P}(k)\backslash\overline{\mathcal{P}}(k)$ gives an optimal augmentation. Hence we stop at an optimal solution. Suppose that $z_k < 0$ for a given $k \in \mathcal{K}$. We illustrate our basis augmentation for three route pairs (r, q_1), (r, q_2) and (r, l_1) in ${S_r}^k$. To cover all possible cases, let us assume that y_{q_1}, y_{q_2} are not part of the basis and y_{l_1} is part of the basis. For simplicity, we denote a subset of ${S_r}^k$ such that $(r,l) \in {S_r}^k$ and y_l is already part of the basis as ${T_r}^k$. For each route pair, say $(r,m) \in {S_r}^k$ firstly we can augment the basis as $\{{x_{(r,m)}}^k, s_2, s_3\}$. Thus our initial basis can be rewritten as

$$B_r = \left(\begin{array}{c|c|c|c|c|c|c|c}
B & X_1 & X_1 & . & X_1 & M & 0 & 0 \\ \hline
0 & D_1 & 0 & . & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & D_1 & . & 0 & 0 & 0 & 0 \\ \hline
. & . & . & . & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & D_1 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & -I & 0 & 0 \\ \hline
G & 0 & 0 & 0 & 0 & -I & -I & 0 \\ \hline
H & 0 & 0 & 0 & 0 & -I & 0 & I
\end{array}\right), \tag{3.44}$$

where $D_1 = \begin{pmatrix} -1 & 0 & 0 \\ -1 & -1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$ and $X_1 = \begin{pmatrix} F & 0 & 0 \end{pmatrix}$. The matrix $M = \begin{pmatrix} F & F & \dots & F \end{pmatrix}$ with the size of $|{T_r}^k|$. Dots imply that there may be other $(r,m) \in {S_r}^k$. To augment the basis easily, similar to Lemma 1, we keep same variables together for the blocks of ${T_r}^k$.

The inverse of the basis is then given by

$$B_r^{-1} = \left(\begin{array}{c|c|c|c|c|c|c|c}
B^{-1} & -B^{-1}X_1D_1^{-1} & -B^{-1}X_1D_1^{-1} & . & -B^{-1}X_1D_1^{-1} & B^{-1}M & 0 & 0 \\ \hline
0 & D_1^{-1} & 0 & . & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & D_1^{-1} & . & 0 & 0 & 0 & 0 \\ \hline
. & . & . & . & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & D_1^{-1} & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & -I & 0 & 0 \\ \hline
GB^{-1} & GB^{-1}X_1D_1^{-1} & GB^{-1}X_1D_1^{-1} & . & . & GB^{-1}M + I & -I & 0 \\ \hline
HB^{-1} & HB^{-1}X_1D_1^{-1} & HB^{-1}X_1D_1^{-1} & . & . & HB^{-1}M + I & 0 & I
\end{array}\right) \tag{3.45}$$

The dual values for the associated matrix are become by

$$
\begin{aligned}
\overline{y}_{aug} &= \begin{pmatrix} c_B & \mathbf{0} & \mathbf{0} & . & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} {B_r}^{-1} \\
&= \begin{pmatrix} c_B B^{-1} & | - c_B B^{-1} X_1 D_1^{-1} + \mathbf{0}\mathbf{D_1^{-1}} & | & ... & | c_B B^{-1} M & |\mathbf{0} & |\mathbf{0} \end{pmatrix} \\
&= \begin{pmatrix} c_B B^{-1} & |Z_1 & | & ... & |Z_1 & |Z_2 & |\mathbf{0} & |\mathbf{0} \end{pmatrix}
\end{aligned}
$$

where $Z_1 = \begin{pmatrix} -z_k & 0 & 0 \end{pmatrix}$ and $Z_2 = \begin{pmatrix} -z_k & -z_k & ... & -z_k \end{pmatrix}$ which has size of $|T_r^k|$. Because of our basis construction, we should pivot y_r firstly. Thus we should calculate the reduced cost of y_r by

$$
\overline{c}_{raug} = c_r - \overline{y}_{aug} \begin{pmatrix} A_r | & 1 & 0 & 1 & ... & \mathbf{1} & \mathbf{0} & \mathbf{1} \end{pmatrix}^T = \overline{c_r} - |{S_r}^k| z_k.
$$

If $\overline{c}_r - |{S_r}^k| z_k < 0$, y_r can enter the basis. Otherwise, our basis is optimal and we can terminate our iterations. Let us assume that it is less than zero. In the next stage, we apply the minimum ratio test to place y_r in basis and calculate $\overline{A}_{raug}$.

To emphasize our stoping condition, we want that s_2 for the first linking constraint leave the basis. Thus we next show that the second entry of the added first block on $\overline{A}_{raug}$, denote by $\overline{A}_{n,r_{aug}}$ has a nonzero value. It should be also emphasized that for the same entry, say $\overline{b}_{naug}$ is zero for the sake of the minimum ratio test. Then we have

$$
\begin{aligned}
\overline{A}_{raug} &= {B_r}^{-1} A_{raug} = {B_r}^{-1} \begin{pmatrix} A_r & | & 1 & 0 & 1 & | & ... & | & \mathbf{1} & \mathbf{0} & \mathbf{1} \end{pmatrix}^T \\
\overline{A}_{n,r_{aug}} &= \begin{pmatrix} 0 & | & 1 & -1 & 0 & | & 0 & 0 & 0 & ... \end{pmatrix} \begin{pmatrix} A_r & | & 1 & 0 & 1 & | & ... & | & \mathbf{1} & \mathbf{0} & \mathbf{1} \end{pmatrix} = 1, \\
\overline{b}_{aug} &= {B_r}^{-1} b_{aug} = {B_r}^{-1} \begin{pmatrix} b & | & 0 & 0 & 1 & | & ... & | & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix}^T \\
\overline{b}_{naug} &= \begin{pmatrix} 0 & | & 1 & -1 & 0 & | & 0 & 0 & 0 & ... \end{pmatrix} \begin{pmatrix} b & | & 0 & 0 & 1 & | & ... & | & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix} = 0.
\end{aligned}
$$

Consequently, y_r can enter the basis at this specific row. Next, we should update B_r and continue our iterations. That is

$$
B_r = \left(\begin{array}{c|c|c|c|c|c|c|c}
B & X_2 & X_1 & . & X_1 & M & 0 & 0 \\ \hline
0 & D_2 & 0 & . & 0 & 0 & 0 & 0 \\ \hline
0 & D_3 & D_1 & . & 0 & 0 & 0 & 0 \\ \hline
. & . & . & . & 0 & 0 & 0 & 0 \\ \hline
0 & D_3 & 0 & 0 & D_1 & 0 & 0 & 0 \\ \hline
0 & D_4 & 0 & 0 & 0 & -I & 0 & 0 \\ \hline
G & 0 & 0 & 0 & 0 & -I & -I & 0 \\ \hline
H & D_4 & 0 & 0 & 0 & -I & 0 & I
\end{array} \right), \tag{3.46}
$$

where $D_2 = \left(\begin{smallmatrix} -1 & 1 & 0 \\ -1 & 0 & 0 \\ -1 & 1 & 1 \end{smallmatrix}\right)$, $D_3 = \left(\begin{smallmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{smallmatrix}\right)$, $D_4 = \left(\begin{smallmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{smallmatrix}\right)$ and $X_2 = \left(\begin{smallmatrix} F & A_r & 0 \end{smallmatrix} \right)$. The inverse of the basis is given by

$$B_r^{-1} = \left(\begin{array}{c|c|c|c|c|c|c|c} B^{-1} & -B^{-1}K & -B^{-1}X_1D_1^{-1} & . & -B^{-1}X_1D_1^{-1} & B^{-1}M & 0 & 0 \\ \hline 0 & D_2^{-1} & 0 & . & 0 & 0 & 0 & 0 \\ \hline 0 & D_3D_2^{-1} & D_1^{-1} & . & 0 & 0 & 0 & 0 \\ \hline . & . & . & . & 0 & 0 & 0 & 0 \\ \hline 0 & D_3D_2^{-1} & 0 & 0 & D_1^{-1} & 0 & 0 & 0 \\ \hline 0 & D_4D_2^{-1} & 0 & 0 & -I & 0 & 0 & 0 \\ \hline GB^{-1} & -GB^{-1}K - D_4D_2^{-1} & GB^{-1}X_1D_1^{-1} & ... & ... & GB^{-1}M + I & -I & 0 \\ \hline HB^{-1} & -HB^{-1}K & HB^{-1}X_1D_1^{-1} & . & HB^{-1}X_1D_1^{-1} & HB^{-1}M + I & 0 & I \end{array}\right), \tag{3.47}$$

where $K = (X_2D_2^{-1} + \sum\limits_{|S_r{}^k|-|T_r{}^k|-1} X_1D_3D_2^{-1} + MD_4D_2^{-1})$. The dual values for the associated matrix become

$$\overline{y}_{aug} = \begin{pmatrix} c_B & c_{B_1} & \mathbf{0} & ... & \mathbf{0} & 0 & 0 & 0 \end{pmatrix} B_r{}^{-1}$$
$$\begin{pmatrix} c_BB^{-1} & | - c_BB^{-1}K + c_{B_1}D_2^{-1} & | - c_BB^{-1}X_1D_1^{-1} & | & ... & | & B^{-1}M & 0 & 0 \end{pmatrix} =$$
$$\begin{pmatrix} c_BB^{-1} & |\overline{c}_r - (|S_r{}^k| - 1)z_k & (|S_r{}^k|)z_k - \overline{c}_r & 0 & |Z_1 & |...| & Z_1 & |Z_2 & |\mathbf{0} & |\mathbf{0} \end{pmatrix}.$$

where $c_{B_1} = \left(\begin{smallmatrix} 0 & c_r & 0 \end{smallmatrix} \right)$. Clearly the dual values of the existing constraints do not change. We next pivot out y_{q_1}. Thus, we calculate the reduced cost of y_{q_1} by

$$\overline{c}_{q_1 aug} = c_{q_1} - \overline{y}_{aug} \begin{pmatrix} A_{q_1} & | & 0 & 1 & 1 & 0 & 0 & ... & 0 \end{pmatrix}^T = \overline{c}_{q_1} + \overline{c}_r - |S_r{}^k| z_k.$$

If $\overline{c}_{q_1} + \overline{c}_r - |S_r{}^k|z_k < 0$, then y_{q_1} enters the basis. Otherwise, our basis is optimal and we can terminate our iterations. Let us assume that it is less than zero. To emphasize our stoping condition, we again force s_2 for the second linking constraint to leave the basis. Thus, we show that the second entry of first block on $\overline{A}_{q_1 aug}$, say $\overline{A}_{n,q_1 aug}$, has nonzero value. It should be also emphasized that for the same entry, $\overline{b}_{naug}$ is zero for the sake of minimum ratio test. That is

$$\overline{A}_{q_1 aug} = B_r{}^{-1}A_{q_1 aug} = B_r{}^{-1} \begin{pmatrix} A_{q_1} & | & 0 & 1 & 1 & | & ... & | & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}^T$$
$$\overline{A}_{n,q_1 aug} = \begin{pmatrix} 0 & | & -1 & 1 & 0 & |1 & -1 & 0 & |... \end{pmatrix} \begin{pmatrix} A_{q_1} & | & 0 & 1 & 1 & | & ... & | & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} = 1,$$

$$\overline{b}_{aug} = B_r{}^{-1}b_{aug} = B_r{}^{-1} \begin{pmatrix} b & | & 0 & 0 & 1 & | & ... & | & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix}^T$$
$$\overline{b}_{naug} = \begin{pmatrix} 0 & | & -1 & 1 & 0 & | & 1 & -1 & 0 & |... \end{pmatrix} \begin{pmatrix} b & | & 0 & 0 & 1 & | & ... & | & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix} = 0.$$

According to the equations above, y_{q_1} can enter the basis at this specific row. Next, we should update B_r and continue our iterations. Then, we have

$$B_r = \left(\begin{array}{c|c|c|c|c|c|c|c}
B & X_2 & X_3 & X_1 & . & M & 0 & 0 \\ \hline
0 & D_2 & D_5 & . & . & 0 & 0 & 0 \\ \hline
0 & D_3 & D_6 & . & . & 0 & 0 & 0 \\ \hline
0 & D_3 & 0 & D_1 & . & 0 & 0 & 0 \\ \hline
. & . & . & . & . & 0 & 0 & 0 \\ \hline
0 & D_4 & 0 & 0 & 0 & -I & 0 & 0 \\ \hline
G & 0 & 0 & 0 & 0 & -I & -I & 0 \\ \hline
H & D_4 & 0 & 0 & 0 & -I & 0 & I
\end{array}\right). \tag{3.48}$$

To take the inverse of the basis, we merge matrices such that $D = \left(\begin{smallmatrix} D_2 & D_5 \\ D_3 & D_6 \end{smallmatrix}\right)$ $X = \left(\begin{smallmatrix} X_2 & X_3 \end{smallmatrix}\right)$ where $D_5 = \left(\begin{smallmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{smallmatrix}\right)$, $D_6 = \left(\begin{smallmatrix} -1 & 0 & 0 \\ -1 & 0 & 0 \\ -1 & 0 & 1 \end{smallmatrix}\right)$ and $X_3 = \left(\begin{smallmatrix} F & A_{q_1} & 0 \end{smallmatrix}\right)$. The inverse of the basis then becomes

$$B_r^{-1} = \left(\begin{array}{c|c|c|c|c|c|c|c}
B^{-1} & -B^{-1}L & -B^{-1}X_1D_1^{-1} & . & -B^{-1}X_1D_1^{-1} & B^{-1}M & 0 & 0 \\ \hline
0 & D^{-1} & 0 & . & 0 & 0 & 0 & 0 \\ \hline
0 & D_3D^{-1} & D_1^{-1} & . & 0 & 0 & 0 & 0 \\ \hline
0 & D_3D^{-1} & 0 & D_1^{-1} & 0 & 0 & 0 & 0 \\ \hline
. & . & . & . & . & 0 & 0 & 0 \\ \hline
0 & D_4D^{-1} & 0 & 0 & -I & 0 & 0 & 0 \\ \hline
GB^{-1} & -GB^{-1}L - D_4D^{-1} & GB^{-1}X_1D_1^{-1} & ... & ... & GB^{-1}F + I & -I & 0 \\ \hline
HB^{-1} & -HB^{-1}L & HB^{-1}X_1D_1^{-1} & ... & ... & HB^{-1}F + I & 0 & I
\end{array}\right) \tag{3.49}$$

where $L = (XD^{-1} + \sum_{|S_r{}^k| - |T_r{}^k| - 2} X_1D_3D^{-1} + MD_4D^{-1})$. The dual values for the associated matrix become

$$\overline{y}_{aug} = \begin{pmatrix} c_B & (c_{B_1}c_{B_1}) & \mathbf{0} & ... \end{pmatrix} B_r{}^{k^{-1}}$$

$$\begin{pmatrix} c_B B^{-1} & | - c_B B^{-1}(XD^{-1} + \sum_{|S_r{}^k| - |T_r{}^k| - 2} X_1D_3D^{-1}) + (c_{B_1}c_{B_1})D^{-1} & | - c_B B^{-1} X_1 D_1^{-1} & |... \end{pmatrix} =$$

$$\begin{pmatrix} c_B B^{-1} & |\overline{c}_{q_1} - z_k & \overline{c}_{q_1} & 0 & -(|S_r{}^k| - 1)z_k + \overline{c}_r + \overline{c}_{q_1} & (|S_r{}^k|)z_k - \overline{c}_r - \overline{c}_{q_1} & 0 & |... \end{pmatrix}$$

Again the dual values associated with the existing constraints do not change. We next pivot out y_{q_2}. Thus, we calculate the reduced cost of y_{q_2} by

$$\overline{c}_{q2\,aug} = c_{q_2} - \overline{y}_{aug} \begin{pmatrix} A_{q_2} & | & 0 & 0 & 0 & 0 & 1 & 1 & ... \end{pmatrix}^T = \overline{c}_{q_2} + \overline{c}_{q_1} + \overline{c}_r - |S_r{}^k| z_k.$$

After illustrating the basis augmentation with the variables that are not already a part of the basis, we generalize our rule for all types of the variables. By following the thinking ahead approach, we never pivot out a variable to the linking constraint set where an already basis variable has nonzero coefficient. For the variables that are already a part of the basis, it is not necessary to do additional iterations. It is clear that, this iterations continue until the last non basic variable enters the basis. In Lemma 2, we have showed that the dual values of the existing constraints are not affected by the existence of some variables already in basis. Since the reduced cost of basic variables equals to zero, we confirm that the basis iterations are correct. For the last variable that enters the basis, the reduced cost value is equal to $\overline{c}_r + \sum_{q \in {S_r}^k} \overline{c}_q - \mid {S_r}^k \mid z_k$. However, according to our assumption we have $\overline{c}_r + \sum_{q \in {S_r}^k} \overline{c}_q - \mid {S_r}^k \mid z_k \geq 0$, and hence we cannot augment the basis. The values of the existing dual variables are also preserved. Thus, continue with pricing problem results with degenerate simplex iteration. It concludes that the current solution is optimal. $\square$

With the lemma above, we prove the optimality of the stopping condition and the validity of the proposed algorithm. To test our algorithm, we next implement a generic code and give some numerical results.

CHAPTER 4

COMPUTATIONAL STUDY

Our solution approach can be analyzed in two dimensions, its practical use and computational success. This chapter includes the details of our implementation. We also present the numerical results obtained by solving a set of randomly generated data.

4.1 Implementation Details

We conduct our computational experiments on a machine with Intel(R) Core(TM)2 Quad CPU and 7.96 GB of RAM running Windows XP. The column generation code is written in Visual C++ and the linear programming relaxation of the MDVSP model is solved by ILOG CPLEX 12.1 using ILOG Concert Technology 2.9. To test our algorithm, we implement the proposed approach in C++. Our program consists of two stages: generating routes and solving the SRMP. Because it makes it easier to represent the network structure and to solve shortest path problems, we use Boost Graph and Date-Time Libraries (see BoostInc. [2011]). To solve the LP models corresponding to each SRMP, we use Cplex 12.1 and Concert Technology (see IBM [2011]).

By means of Boost Graph Libraries, we create a directed graph, which includes the trips(nodes) and connections(arcs). Since each trip has same specific properties like Departure Time, Arrival Time, Departure City, Arrival City, Trip Number, Dual Value and each connection has Deadheading Cost, Head Trip, Tail Trip, we create $Trip$ and $Connection$ classes. Moreover, our problem needs to generate routes, we create another class *Route* that keeps the sequence of trips and has properties like Total Cost, Number of Trips, Depot Number. For the extra trip problem, we also have $ExtraTrip$ class, which has similar properties as the $Trip$ class. For the delayed trip problem, we generate $DelayedTrip$ class, whose properties are Trip Number and the Delay Amount. To represent the arrival and departure times, we use Boost Date-Time Libraries.

According to the proposed method, it is also necessary to generate set ${S_r}^k$ for each

$r \in \mathcal{R}\backslash\overline{\mathcal{R}}$. Since, this subproblem required in the worst case the total enumeration of all routes, we propose another method to solve the problem. At each subproblem, instead of generating whole pairs for r, we just generate a route q such that $\overline{c}_q$ is minimum and our stopping condition simply becomes $\overline{c}_r + \overline{c}_q - z_k$. If $\overline{c}_r + \overline{c}_q - z_k < 0$, we add the route pair to SRMP, otherwise we terminate our algorithm. With this heuristic method, we were able to obtain the optimal solutions for most of the problems and solve problems with as many as 800 trips.

4.2 Numerical Results

We try to solve the MDVSP with disruptions. Depending on the magnitude of the disruption, generally, the conventional model's solution cannot absorb the alterations to the schedule, and hence a considerable number of changes may be required with an increase the cost. In Figure 4.1, the conventional model's solution is illustrated. Three routes cover the 12 trip, and the total operation cost in terms of deadheading cost is calculated as 500. All other deadheading costs are zero. Unfortunately, as predicted before, trip 5 is delayed and it reaches the arrival city at 12:30. This tiny disruption delays all the following trips in that route.

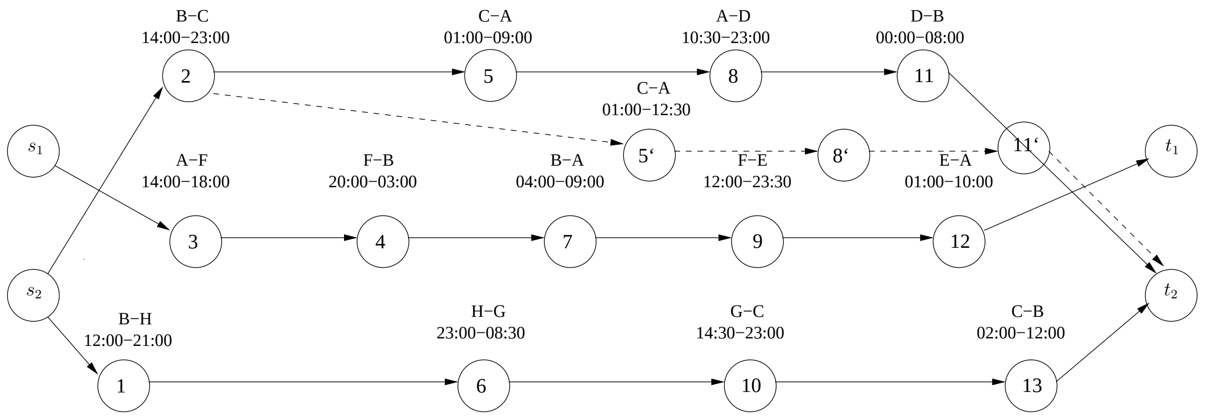


Figure 4.1: Consequences of the delay on the sample network when the conventional model is used

In Figure 4.1 when trip 5 is delayed, an available vehicle is waiting for its following duty which is later than the departure time of trip 8. This vehicle can be assigned to trip 8, however, its following trip should be covered by a vehicle. At operational level, not to delay succeeding trips, a chain of swaps needs to be executed. A group of vehicles swaps their duties among them and the disruption can be handled as illustrated in Figure 4.2.

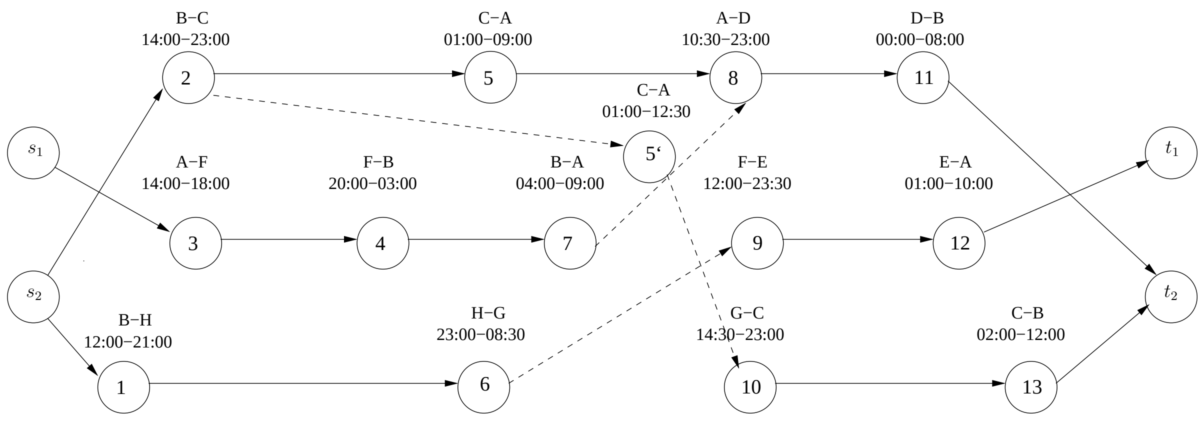


Figure 4.2: Chain of swaps required to handle the delay when the solution of conventional method is used

In Figure 4.2 the schedules of three vehicles are changed but this solution does not work and results in a more complex situation because to execute this solution, an extra free vehicle is required. Instead, with our solution method, we obtain a set of routes that includes possible swaps for a disruption. This kind of solution may have higher operational cost than the conventional one but allows flexibility to absorb disruptions.

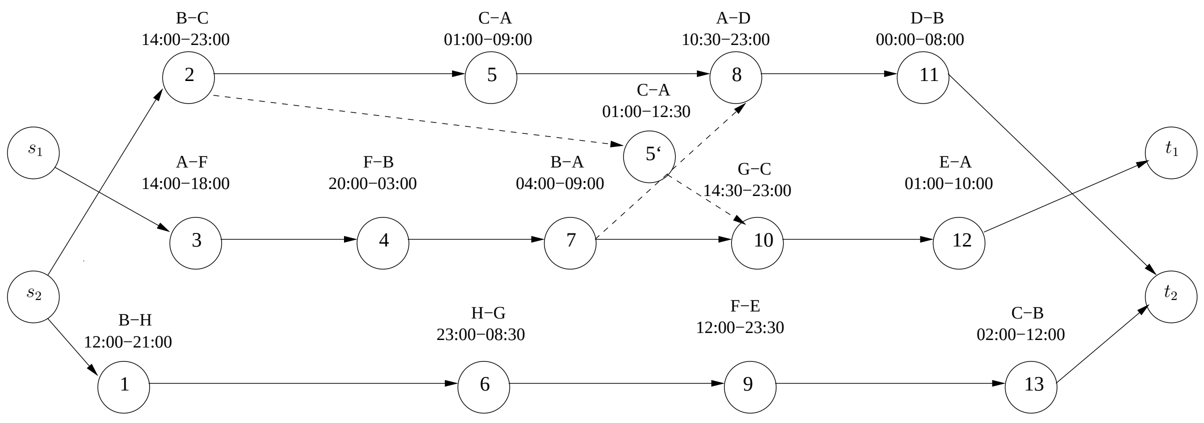


Figure 4.3: The proposed solution of problem on the sample network

In Figure 4.3, the proposed solution for the sample network is given. Some routes are designed different than Figure 4.2 to handle the delay. Since the delayed vehicle can cover trip 8, this trip becomes a part of the second route. However, the total cost now becomes 700. It is clear that the proposed method can create a simple solution for each possible delayed trip.

To test our method's computational efficiency, we next consider several problem instances with different sizes. We generate data for timetabled trips and disruptions. We select 33 cities in Turkey as our locations. We also determine five of these cities as possible base locations. The connection distances data between cities are collected from the web page of Turkey General Directorate of Highways (see GDH [2011]). The travel durations between the cities are directly derived from that data. Similar to the random data generation mechanism of Carpaneto et al. [1989a], we determine the arrival and departure locations of each trip by sampling from uniform distribution. We also determine the departure time of each trip uniformly. We assume that each trip can depart every half an hour.

For the extra trip problem, we first determine the set of cities that extra trip may depart and arrive. From this set, we determine the extra trips similar to timetabled trips.For delayed trip problem, we again choose the trip that can possibly arrive late from the set of timetabled trips randomly. Their delay amount is also uniformly distributed between 60 and 180 minutes. To determine the capacity of each depot, we use the idea of Carpaneto et al. [1989b]. However, we also include the total number of depots. For problem with n time tabled trips and m depots, we determine the size of each depot by sampling from uniform distribution with bounds $\left[\frac{n}{2m}, \frac{n}{3m}\right]$. To test our model we determine the number of trips as 100, 200, 300, 500 and 800. The number of depots are between 1 and 5. Up to 20 disruptions are used in our instances.

Recall that we solve the linear relaxation of the overall problem. Therefore, some instances may result with fractional solutions. To obtain integer solution, we solve the existing SRMP with the integer solver of Cplex. In the tables below, both of the LP relaxation and the integer solutions are represented.

Before testing the results obtained with the proposed solution method, we tried to solve the problem without column generation. To test our results, we generate all possible routes and as well as the set that includes all possible swapping solutions for each disruption. Since the size of the routes set and the swapping solution set are really large, only a limited number of problems can be solved with this method.

Table 4.1 includes the computational results for the problem with 100 trips. A variety of instances is provided by different number of depots and disruptions. Note that $n-m-k$ represent the number of trips, depots and disruptions respectively in the first column. The second column of the table shows the LP objective function value for each instance which is same for both of the proposed and conventional method. The third column represent the IP solution of our proposed method. The fourth column includes the total computation time of the proposed method combined with the solution time of IP solver, when the solution is fractional. The last two column represents the IP solution and computation time for the all columns generation strategy. The last column represents the computational time difference between proposed method and the conventional method. We can emphasize that for smaller instances, conventional method works faster. However, when the problem gets larger, our proposed method is better. We also note that the average optimality gap using column-and-row generation in a heuristic way is only 0.8 % for these 25 instances.

Clearly, when the number of disruptions and depots increase, the total number of routes and the total number of swapping solutions increase dramatically, In the case of 5 disruptions and 5 depots, 118,995 swapping solutions are available. According to the mathematical model, each swapping solution has three linking constraint, so that the total number of rows is larger than 350,000. This numbers show that it is very difficult to solve large instances. As last column shows, it is easier to solve smaller instances by generating all routes. However, when the problem becomes large, the proposed algorithm seems to be the only option. In case of 5 disruptions and 5 depots, the computation time difference between them is approximately 1,308%.

Table 4.1: Comparison of all routes generation and the proposed method

Instance size	**OFV of proposed method (LP)**	**OFV of proposed method (IP)**	**Total CPU time of proposed method (IP)**	**OFV of conventional method (IP)**	**CPU time of conventional method (IP)**	**Decrease in Comp. time (%)**
100-1-1	34,380.00	34,380.00	6.094	34,380.00	1.989	-67.36
100-1-2	34,380.00	34,380.00	5.953	34,380.00	2.543	-57.28
100-1-3	35,055.00	35,055.00	6.235	35,055.00	2.693	-56.81
100-1-4	35,055.00	35,055.00	7.562	35,055.00	2.661	-64.81
100-1-5	35,055.00	35,055.00	7.984	35,055.00	3.814	-52.23
100-2-1	28,605.00	28,605.00	6.907	28,605.00	7.177	3.91
100-2-2	28,605.00	28,605.00	7.454	28,605.00	8.001	7.34
100-2-3	28,725.00	28,725.00	7.516	28,725.00	10.738	42.87
100-2-4	28,725.00	28,725.00	8.407	28,725.00	12.045	43.27
100-2-5	28,725.00	28,725.00	9.485	28,725.00	15.396	62.32
100-3-1	22,095.00	22,095.00	9.454	22,095.00	31.53	233.51
100-3-2	22,095.00	22,755.00	9.923	22,755.00	60.005	504.71
100-3-3	22,102.50	22,755.00	10.688	22,755.00	73.69	589.46
100-3-4	22,102.50	22,755.00	11.844	22,755.00	90.225	661.78
100-3-5	22,102.50	22,755.00	12.798	22,590.00	106.98	735.91
100-4-1	20,670.00	20,670.00	12.969	20,670.00	85.883	562.22
100-4-2	20,670.00	20,670.00	13.516	20,670.00	99.37	635.2
100-4-3	20,670.00	20,670.00	14.656	20,670.00	106.725	628.2
100-4-4	20,910.00	20,910.00	15.61	20,670.00	110.785	609.71
100-4-5	20,910.00	20,910.00	16.454	20,910.00	132.835	707.31
100-5-1	20,910.00	20,910.00	16.157	20,910.00	103.06	537.87
100-5-2	20,910.00	20,910.00	16.657	20,910.00	129.181	675.54
100-5-3	20,910.00	22,185.00	17.391	20,910.00	166.178	855.54
100-5-4	21,240.00	22,515.00	18.938	20,910.00	245.943	1198.67
100-5-5	21,127.50	22,620.00	19.548	21,550.00	419.227	2044.6

Table 4.2: Results for large instances of the delayed trip problem

Instance size	**OFV of LP solution**	**CPU time of LP solution**	**OFV of IP solution**	**CPU time of IP solution**	**Difference between OFV of LP and IP solutions(%)**
200-1-1	62445.0	14.281	62445	0.062	0.0000
200-1-2	62445.0	14.141	62445	0.093	0.0000
200-1-3	62445.0	15.313	62445	0.078	0.0000
200-1-4	62445.0	15.484	62445	0.078	0.0000
200-1-5	44445.0	30.985	44445	0.125	0.0000
200-2-1	44467.5	37.156	45405	0.093	0.0211
200-2-2	44737.5	32.797	44745	0.094	0.0002
200-2-3	45772.5	36.813	46140	0.109	0.0080
200-2-4	46572.5	35.094	46800	0.094	0.0049
200-2-5	62805.0	14.765	63165	0.093	0.0057
200-3-1	38535.0	37.828	38535	0.078	0.0000
200-3-2	38535.0	40.515	38535	0.094	0.0000
200-3-3	38955.0	42.312	39375	0.109	0.0108
200-3-4	39045.0	36.484	39045	0.078	0.0000
200-3-5	39281.3	42.109	39525	0.109	0.0062
200-4-1	36270.0	37.953	36270	0.094	0.0000
200-4-2	36270.0	40.875	36270	0.094	0.0000
200-4-3	36780.0	37.422	36780	0.093	0.0000
200-4-4	36900.0	43.204	37110	0.094	0.0057
200-4-5	39478.8	44.250	41550	0.109	0.0525
200-5-1	35745.0	41.515	35745	0.094	0.0000
200-5-2	36255.0	40.765	36255	0.094	0.0000
200-5-3	36690.0	49.469	37110	0.125	0.0114
200-5-4	37605.0	47.640	38280	0.188	0.0179
200-5-5	39577.5	45.359	39900	0.110	0.0081
300-1-1	96435.0	32.687	96435	0.078	0.0000
300-1-2	96435.0	34.000	96435	0.094	0.0000
300-1-3	96585.0	35.062	96735	0.219	0.0016
300-1-4	96975.0	36.360	98145	0.078	0.0121
300-1-5	97065.0	37.281	97300	0.125	0.0024
300-2-1	69610.0	102.551	69810	0.203	0.0029
300-2-2	69660.0	59.549	69810	0.125	0.0022
300-2-3	69660.0	60.643	69810	0.125	0.0022
300-2-4	69660.0	63.206	69810	0.125	0.0022
300-2-5	69954.4	67.065	70170	0.125	0.0031
300-3-1	59160.0	135.484	59160	0.125	0.0000
300-3-2	59520.0	141.984	59880	0.156	0.0060
300-3-3	59670.0	126.515	59670	0.157	0.0000
300-3-4	59910.0	147.125	60540	0.109	0.0105
300-3-5	60207.5	144.078	60480	0.156	0.0045
300-4-1	55095.0	139.125	55455	0.203	0.0065
300-4-2	55177.5	133.453	55245	0.172	0.0012
300-4-3	55635.0	151.141	56100	0.203	0.0084
300-4-4	55961.3	134.891	57930	0.171	0.0352
300-4-5	56360.0	146.828	56760	0.219	0.0071
300-5-1	53265.0	136.453	53475	0.125	0.0039
300-5-2	53422.5	137.531	53790	0.187	0.0069
300-5-3	53565.0	135.859	53565	0.188	0.0000
300-5-4	54330.0	150.750	54960	0.125	0.0116
300-5-5	55584.2	146.406	57735	0.204	0.0387
500-1-1	163020.0	124.453	163020	0.437	0.0000
500-1-2	163020.0	128.421	163020	0.266	0.0000
500-1-3	163170.0	129.843	163320	0.281	0.0009
500-1-4	163650.0	133.515	163650	0.219	0.0000
500-1-5	163740.0	133.656	164100	0.281	0.0022
500-2-1	131685.0	202.625	131685	0.172	0.0000
500-2-2	131745.0	219.219	131745	0.281	0.0000
500-2-3	132159.0	235.594	132810	0.203	0.0049
500-2-4	132165.0	330.375	132210	0.172	0.0003
500-2-5	132608.0	235.656	133080	0.219	0.0036
500-3-1	114780.0	278.750	114810	0.188	0.0003
500-3-2	114983.0	297.657	115200	0.312	0.0019
500-3-3	115028.0	282.937	115035	0.344	0.0001
500-3-4	115673.0	334.735	116190	0.203	0.0045
500-3-5	117820.0	296.891	117975	0.218	0.0013
500-4-1	112200.0	295.453	112200	0.172	0.0000
500-4-2	112428.0	297.141	112695	0.281	0.0024
500-4-3	113520.0	307.765	114780	0.219	0.0111
500-4-4	114135.0	314.062	114165	0.171	0.0003
500-4-5	115245.0	304.297	115245	0.219	0.0000
500-5-1	100538.0	354.469	100665	0.344	0.0013
500-5-2	100545.0	337.593	100785	0.250	0.0024
500-5-3	100620.0	336.078	100620	0.219	0.0000
500-5-4	101968.0	363.406	103080	0.375	0.0109
500-5-5	102818.0	368.047	104175	0.219	0.0132
800-1-1	253560.0	609.781	253560	0.609	0.0000
800-1-2	253560.0	622.610	253560	0.469	0.0000
800-1-3	253560.0	629.750	253560	0.610	0.0000
800-1-4	253875.0	630.390	254190	0.812	0.0012
800-1-5	253920.0	631.078	254280	0.547	0.0014
800-2-1	245955.0	507.594	245955	0.734	0.0000
800-2-2	245955.0	700.484	245955	0.500	0.0000
800-2-3	245955.0	520.296	245955	0.047	0.0000
800-2-4	245955.0	505.515	245955	0.453	0.0000
800-2-5	246630.0	506.484	246630	0.703	0.0000
800-3-1	242910.0	510.109	242910	0.500	0.0000
800-3-2	242910.0	513.531	242910	0.484	0.0000
800-3-3	242910.0	515.985	242910	0.062	0.0000
800-3-4	242910.0	522.891	242910	0.328	0.0000
800-3-5	243225.0	519.437	243540	0.532	0.0013

Table 4.3: Results for large instances of the extra trip problem

Instance size	OFV of LP solution	CPU time of LP solution	OFV of IP solution	CPU time of IP solution	Difference between OFV of LP and IP solutions(%)
200-1-1	62445.0	17.281	62745	0.094	0.0048
200-1-2	62445.0	18.656	62745	0.094	0.0048
200-1-3	62445.0	20.516	62445	0.094	0.0000
200-1-4	62445.0	22.625	62445	0.094	0.0000
200-1-5	63735.0	30.297	63735	0.140	0.0000
200-2-1	44205.0	37.438	44205	0.078	0.0000
200-2-2	44670.0	52.016	44835	0.110	0.0037
200-2-3	44790.0	41.359	45090	0.078	0.0067
200-2-4	45090.0	43.250	45090	0.109	0.0000
200-2-5	45650.3	62.609	45830	0.109	0.0039
200-3-1	38535.0	42.375	38535	0.125	0.0000
200-3-2	38535.0	45.891	39810	0.094	0.0331
200-3-3	38535.0	51.406	39810	0.110	0.0331
200-3-4	38535.0	55.000	39810	0.015	0.0331
200-3-5	39330.0	72.219	39330	0.109	0.0000
200-4-1	36270.0	43.703	36270	0.094	0.0000
200-4-2	36270.0	46.765	37545	0.110	0.0352
200-4-3	36270.0	51.687	36930	0.110	0.0182
200-4-4	37065.0	57.953	37065	0.032	0.0000
200-4-5	37065.0	73.484	37065	0.094	0.0000
200-5-1	35745.0	47.938	35745	0.109	0.0000
200-5-2	35745.0	53.344	35745	0.093	0.0000
200-5-3	35745.0	58.156	35745	0.110	0.0000
200-5-4	36420.0	62.719	36420	0.094	0.0000
200-5-5	38228.8	83.282	39960	0.140	0.0453
300-1-1	96435.0	40.750	96435	0.109	0.0000
300-1-2	96435.0	41.906	96435	0.079	0.0000
300-1-3	96435.0	45.016	96435	0.094	0.0000
300-1-4	97380.0	49.610	97380	0.063	0.0000
300-1-5	97380.0	58.859	97380	0.141	0.0000
300-2-1	69510.0	60.424	69510	0.078	0.0000
300-2-2	69810.0	64.675	69810	0.094	0.0000
300-2-3	69810.0	63.378	69810	0.109	0.0000
300-2-4	69810.0	65.253	69810	0.078	0.0000
300-2-5	69909.0	69.862	70125	0.078	0.0031
300-3-1	59160.0	150.219	59160	0.125	0.0000
300-3-2	59160.0	156.000	60450	0.141	0.0218
300-3-3	59160.0	166.375	60450	0.156	0.0218
300-3-4	60285.0	180.047	63360	0.328	0.0510
300-3-5	61260.0	208.125	61260	0.172	0.0000
300-4-1	54735.0	153.687	54735	0.141	0.0000
300-4-2	54735.0	156.187	56010	0.141	0.0233
300-4-3	54735.0	166.453	56010	0.156	0.0233
300-4-4	55410.0	172.828	55410	0.032	0.0000
300-4-5	55410.0	212.515	55410	0.157	0.0000
300-5-1	53055.0	156.734	53055	0.109	0.0000
300-5-2	53055.0	159.578	54330	0.141	0.0240
300-5-3	53055.0	169.296	54330	0.157	0.0240
300-5-4	54525.0	176.547	54705	0.047	0.0033
300-5-5	54525.0	216.110	54705	0.172	0.0033
500-1-1	163020.0	137.735	163020	0.375	0.0000
500-1-2	163020.0	143.203	163020	0.218	0.0000
500-1-3	163020.0	148.563	163020	0.266	0.0000
500-1-4	163020.0	160.468	163020	0.250	0.0000
500-1-5	163650.0	172.750	163650	0.219	0.0000
500-2-1	131670.0	227.156	131670	0.219	0.0000
500-2-2	131745.0	251.094	131745	0.187	0.0000
500-2-3	131745.0	358.031	131745	0.281	0.0000
500-2-4	132608.0	268.844	133080	0.063	0.0036
500-2-5	132608.0	296.625	133080	0.187	0.0036
500-3-1	115380.0	292.735	115575	0.218	0.0017
500-3-2	115455.0	299.031	115875	0.203	0.0036
500-3-3	115639.0	339.281	116190	0.265	0.0048
500-3-4	115703.0	300.016	115875	0.235	0.0015
500-3-5	116109.0	353.703	116580	0.234	0.0041
500-4-1	95910.0	1432.450	95910	1.329	0.0000
500-4-2	95910.0	468.750	95910	0.219	0.0000
500-4-3	95910.0	916.672	97185	0.625	0.0133
500-4-4	96397.5	788.297	97245	0.594	0.0088
500-4-5	96705.0	918.234	96705	0.625	0.0000
500-5-1	93090.0	743.734	93090	0.344	0.0000
500-5-2	93090.0	706.390	93090	0.422	0.0000
500-5-3	93607.5	792.422	95790	0.828	0.0233
500-5-4	94057.5	757.047	94380	0.453	0.0034
500-5-5	94245.0	1038.390	94695	0.594	0.0048
800-1-1	253560.0	609.781	253560	0.609	0.0000
800-1-2	253560.0	622.610	253560	0.469	0.0000
800-1-3	253560.0	629.750	253560	0.610	0.0000
800-1-4	253810.0	631.078	254280	0.812	0.0019
800-1-5	254190.0	630.390	254190	0.547	0.0000
800-2-1	245955.0	507.594	245955	0.734	0.0000
800-2-2	245955.0	700.484	245955	0.500	0.0000
800-2-3	245955.0	520.296	245955	0.047	0.0000
800-2-4	245955.0	505.515	245955	0.453	0.0000
800-2-5	246630.0	506.484	246630	0.703	0.0000
800-3-1	219330.0	1107.550	219330	0.844	0.0000
800-3-2	219330.0	1327.280	219330	2.531	0.0000
800-3-3	219330.0	1147.200	219330	2.172	0.0000
800-3-4	221455.5	1150.500	221900	2.435	0.0020
800-3-5	221455.5	1212.830	221900	2.768	0.0020

In Table 4.2 and Table 4.3, the computational results for large instances of the delayed trip problem and the extra trip problem are given respectively. The second and the third columns include the objective function values and the computation times for linear relaxation solutions, respectively. Next two columns include results for integer solutions. The last column gives the gap between the objective function values of the linear and integer solutions. According to our computational study, we observe that most of the time, the linear relaxation and integer solutions have the same results.

The solutions with large instances show that up to 800 trips, 5-disruptions instances can be solved easily. However, for 800 trip problem up to 3-depot instances can be solved. Table 4.2 and Table 4.3 illustrate that, when the total number of trips increases, the total deadheading cost also increases. Since, adding a new depot with extra capacity relaxes the existing solution, it also decreases the objective function value. However, obtaining the optimal solution becomes much more difficult and the computation time increases.

To investigate the effect of the number of disruptions, we add up to 20 disruptions to a selected problem with 300 trips and 2 depots. In Table 4.4, these results are reported for both disruption types.

Table 4.4: Results for varying numbers of disruptions (m=300)

Instance size	**OFV change for extra trip** (%)	**Comp. Time for extra trip**	**OFV change for delayed trip**(%)	**Comp. Time for delayed trip**
300-3-0	0.00000	58.533	0.00000	59.533
300-3-1	0.00432	62.754	0.00000	60.502
300-3-2	0.00432	59.674	0.00432	64.769
300-3-3	0.00432	60.768	0.00432	63.487
300-3-4	0.00432	63.331	0.00432	65.331
300-3-5	0.00950	67.19	0.00885	69.94
300-3-6	0.02460	66.518	0.00885	69.284
300-3-7	0.02611	69.659	0.00885	94.847
300-3-8	0.02698	70.894	0.00885	88.691
300-3-9	0.03388	76.035	0.00885	77.815
300-3-10	0.03841	105.16	0.00885	78.518
300-3-11	0.03971	78.112	0.00885	81.722
300-3-12	0.04769	76.363	0.00885	84.738
300-3-13	0.04834	73.785	0.03388	85.566
300-3-14	0.05050	81.254	0.03388	99.691
300-3-15	0.05179	82.503	0.03388	127.302
300-3-16	0.06452	79.737	0.03798	94.019
300-3-17	0.07531	83.847	0.04834	96.939
300-3-18	0.07574	141.412	0.04834	94.696
300-3-19	0.07574	92.931	0.05050	98.516
300-3-20	0.07574	97.836	0.06452	98.329

As a reference point, we solve the problem without any disruptions. In Table 4.4, the second column is the increase in the objective function value for the problem when the disruption is due to an extra trip. The third row is the computation time for this problem. Next two columns represent the similar results for the delayed trip problem. We emphasize that our objective function comparison includes the objective function values of the integer solutions. According to the results given at Table 4.4, adding a new disruption may increases the cost but sometimes no change is observed. This stability of objective function value is caused by alternate optimal solutions that can also cover the newly added disruption. If we investigate the results in terms of the types of disruptions, it can be observed that the schedule is more vulnerable to emergence of extra trips because for most of the entries in the second column are greater than the entries in the fourth column.

We should also emphasize that in the case of 20 disruption are take into consideration, the objective function value of integer solution increases approximately 7.5%. However, we note once again that if the proposed solution method is not used, it may not be possible for the company to handle these disruptions with the conventional MDVSP solution. Therefore, a more significant increase in the cost may be observed.

Figure 4.4 illustrates the relationship between the number of disruptions and the objective function value changes for a problem with 500 trips. For the instances with larger number of depots (represented by m in figure) adding more disruptions has more affect on the objective function values.

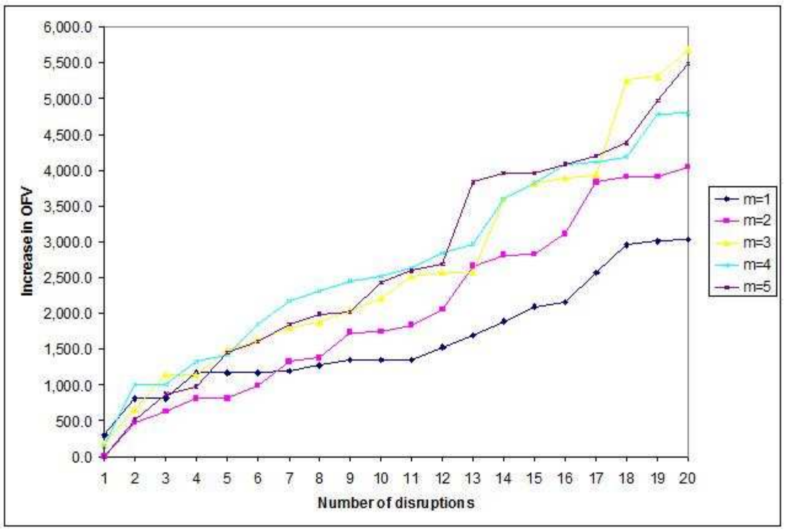


Figure 4.4: Objective function value increase when adding numbers of disruptions for variety of depot size

CHAPTER 5

CONCLUSION

Because of uncertainties that are inherent in the duration of trips and customer demand, the solution obtained by solving the traditional vehicle scheduling problem usually cannot satisfy the requirements of the public transportation companies.

In this thesis, we solve the multi-depot vehicle scheduling problem with disruptions. Specifically, we focus on the extra trips and the delayed trips. To solve this extension of MDVSP, we propose a method which is based on simultaneous column-and-row generation. To generate columns and rows, we use two different pricing subproblems. One of these subproblems individually generates routes and the other one generates pairs of routes. These subproblems lead to specific versions of the shortest path problem. We also show that the algorithm converges to the optimal solution.

We implement the proposed algorithm using generic programming tools. To test our method, we randomly generate problem instances up to 800 nodes for both delayed trip and extra trip problems. As a benchmark strategy, we generate the complete set of routes for small problems and solve them to optimality by standard linear programming and integer programming solvers. The conventional method cannot deal with large instances, however, the proposed method is able to solve these instances in a reasonable time. According to our computational results, the deadheading cost increases only slightly even when a large number of disruptions exist. The conducted computational study shows that the proposed method both computationally and practically efficient.

Bibliography

Bertossi, A. A., Carraresi, P., and Gallo, G. (1987). On some matching problems arising in vehicle scheduling models. *Networks*, 17:271–281.

Bodin, L., Golden, B., Assad, A., and Ball, M. (1983). Routing and scheduling of vehicles and crews: The state of the art. *Computers and Operations Research*, 10:63–211.

BoostInc. (2011). Boost c++ libraries. `http://www.boost.org/`.

Carpaneto, D. M., Dell Amico, M., Fischetti, M., and Toth, P. (1989a). A branch and bound algorithm for the multiple vehicle scheduling problem. *Networks*, 19(5):531–548.

Carpaneto, G., Dell'Amico, M., Fischetti, M., and P., T. (1989b). A branch-and-cut algorithm for the multiple depot vehicle scheduling problem. *Networks*, 19:531–548.

Daduna, J., Branco, I., and Paixão, J. M. P. (1993). Vehicle scheduling for public mass transit- an overview. In *Computer-aided transit scheduling Proceedings of the Sixth International Workshop, Lisbon.*

Dantzig, G. and Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research*, 8(1):101–111.

Dell'Amico, M., Fischetti, M., and Toth, P. (1993). Heuristic algorithms for the multiple depot vehicle scheduling problem. *Management Science*, 39(1):115–125.

Desrochers, M. and Soumis, F. (1989). A column generation approach to the urban transit crew scheduling problem. *Transportation Science*, 23(1):1–13.

Feillet, D., Gendreau, M., Medaglia, A., and Walteros, J. (2010). A note on branch-and-cut-and-price. *Operations Research Letters*, 38(5):346–353.

Freling, R., Wagelmans, A. P. M., and Paixão, J. M. P. (2001). Models and algorithms for single-depot vehicle scheduling. *Transportation Science*, 35(2):165–180.

GDH (2011). General directorate of highways webpage. `http://www.kgm.gov.tr`.

Hadjar, A., Marcotte, O., and Soumis, F. (2006). A branch-and-cut algorithm for the multiple depot vehicle scheduling problem. *Operations Research*, 54(1):130–149.

Haghani, A. and Banihashemi, M. (2002). Heuristic approaches for solving large-scale bus transit vehicle scheduling problem with route time constraints. *Transportation Research Part A: Policy and Practice*, 36(4):309–333.

Huisman, D., Freling, R., and Wagelmans, A. (2004). A robust solution approach to the dynamic vehicle scheduling problem. *Transportation Science*, 37(4):447–458.

IBM (2011). Ilog software. `http://www.ilog.com/`.

Kramkowski, S., Kliewer, N., and Meier, C. (2009). Heuristic methods for increasing delay-tolerance of vehicle schedules in public bus transport. In *The VIII Metaheuristics International Conference*, Hamburg, Germany.

Löbel, A. (1998). Vehicle scheduling in public transit and lagrangian pricing. *Management Science*, 44:1637–1649.

Li, J. Q., Borensteinb, D., and Mirchandania, P. B. (2008). A lagrangian heuristic for the real-time vehicle rescheduling problem. *Transportation Research Part E*, 45(3):419–433.

Muter, İ., Birbil, Ş. İ., and Bülbül, K. (2011). Simultaneous column-and-row generation for large-scale linear programs with column-dependent-rows. `http://www.optimization-online.org/DB_HTML/2010/11/2815.html`.

Muter, İ., Birbil, Ş. İ., Bülbül, K., Şahin, G., Taş, D., Tüzün, D., and Yenigün, H. (2010). Solving a robust airline crew pairing problem with column generation. *C*omputers and Operations Research, doi:10.1016/j.cor.2010.11.005.

Ribeiro, C. and Soumis, F. (1994). A column generation approach to the multi depot vehicle scheduling problem. *Operations Research*, 42(1):41–52.

Sato, T., Sakikawa, S., Morita, T., Ueki, N., and Muratay, T. (2009). Crew and vehicle rescheduling based on a network flow model and its application to a railway train operation. In *IAENG International Journal of Applied Mathematics*, volume 39(3), pages 61–74.

Shebalov, S. and Klabjan, D. (2006). Robust airline crew pairing: Move-up crews. *Transportation Science*, 40(3):300–312.

Tekiner, H., Birbil, Ş. İ., and Bülbül, K. (2009). Robust crew pairing for managing extra flights. *Computers and Operations Research*, 36:2031–2048.