A BRANCH-AND-PRICE ALGORITHM FOR RESOURCE CONSTRAINED VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

by

Neda Tanoumand

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

August, 2017

A BRANCH AND PRICE ALGORITHM FOR A RESOURCE-CONSTRAINED VEHICLE ROUTING PROBLEM

APPROVED BY:

Assoc. Prof. Dr. Tonguç Ünlüyurt
(Thesis Supervisor)

[illegible]

Prof. Dr. Bülent Çatay
(Jury Member)

[illegible]

Assoc. Prof. Dr. Selçuk Çolak
(Jury Member)

[illegible]

DATE OF APPROVAL: 01/08/2017

© Neda Tanoumand 2017

All Rights Reserved

A BRANCH-AND-PRICE ALGORITHM FOR RESOURCE CONSTRAINED VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

Neda Tanoumand

Master of Science in Industrial Engineering

Thesis Supervisor: Tonguç Ünlüyurt

Abstract

In this thesis, we consider a variation of vehicle routing problem where different types of resources are required. The problem is motivated by an application for a Home Health Care service provider. In this problem, services are provided by a limited number of personnel (nurses and health care aids) in patients' home. Each patient requires either a nurse or a health aid or both depending on their conditions during a strict time window. The personnel are transported to patients' home by vehicles that can carry at most two people. We assume that a health aid provider cannot be substituted by a nurse and vice versa. The problem can be generalized to cases where patients require different resources at different levels. In this study, a Branch and Price algorithm is implemented to optimally solve the problem. The problem is formulated as a set-partitioning problem and solved by a branch-and-price algorithm. We investigate the efficacy of some implementation features by conducting an extensive computational studies. The computational results show that the efficient exact method surpass the off-the-shelf mixed integer programming solvers.

Keywords: Exact Methods, Home Health Care Routing Problem, Vehicle Routing Problem.

KAYNAK KISITLI ARAÇ ROTALAMA PROBLEMİ İÇİN DAL-VE-FİYAT ALGORİTMASI

Neda Tanoumand

Endüstri Mühendisliği Yüksek Lisansı

Tez Danışmanı: Tonguç Ünlüyurt

Özet

Bu tezde farklı kaynak kullanımının gerekli olduğu, bir Araç Rotalama Problemi çeşidi sunulmuştur. Problem, Evde Bakım Servis sağlayıcıları ile ilgili bir uygulamadan yola çıkılarak tasarlanmıştır. Ayrıca, problemde sağlık hizmetleri, hemşireler ve hasta bakıcılar tarafından hastaların evlerinde sağlanmaktadır. Her hasta, belirli ve değiştirilemeyen zaman aralıklarına bağlı olarak hemşire ve/veya hasta bakıcıya ihtiyaç duyabilir. Sağlık personelleri, hastaların evlerine en fazla iki kişi kapasitesi olan özel araçlar ile ulaştırılmaktadır. Ek olarak, model kurulumunda, hemşire ve hasta bakıcıların birbirlerinin yerine kullanılamayacağı varsayılmıştır. Problem, genel olarak her hastanın farklı sayıda, farklı kaynaklara ihtiyaç duyması ile tanımlanabilir. Öncelikle, problem küme bölme metodu ile modellenmiş, daha sonra en iyilenmesi için Dal-ve-Fiyat algoritması kullanılmıştır. Algoritmanın etkinliğinin ölçülmesi için bazı özellikleri test edilmiştir. Test sonuçları, uygulanan algoritmanın Karışık Tam Sayılı Programlama metodlarından daha iyi çalıştığını göstermektedir.

Anahtar Kelimeler: Araç Rotalama Problemi, Evde Bakım Servis, Pekin Yöntemler

To my best friend . . .

Acknowledgments

I would like to express my sincere gratitude to my thesis supervisor Tonguç Ünlüyurt for his help and encouragement during the course of my master's thesis.

I want to sincerely thank Farzin Asghari Arpatappeh, Gizem Özbaygın, Nozir Shokirov for their valuable contributions to this research. I am also thankful to my best friends Sinem Duman and Sinan Seymen for their encouragements.

I gratefully acknowledge Sabancı University for the scholarships received and for becoming part of my family for last two years.

Contents

List of Figures

List of Tables

1 Introduction

We consider a generalization of the vehicle routing problem with time window (VRPTW) where each customer requires a set of resources. It is not possible to substitute a certain type of resource by another type. This corresponds to different types of resources providing different type of services. This problem is motivated by a home health care service (HHCS) routing problem. In particular, we have 2 types of resources and each patient may require at most one of each type.

HHCS provider companies have become more popular in last decades. These companies cover a broad range of services that are required by care-dependent people at their home. The limited number of beds in the hospitals, the aging in the population, and increasing number of people suffering from chronic disease are some of the factors that lead to the birth of Home Health Care (HHC) companies. HHCSs were originally introduced by an American professor, Bluestone, in 1945. Then, these services were developed in Canada, France, Denmark, etc [1]. In the early age of HHCS, it was comprehended as a mean to make earlier hospital discharge easier. Nowadays, these services are provided for old people, adults with functional limitations, and patients with chronic diseases in the form of long-term or short term care services [2].

There are two different services that HHCS companies are providing, Nursing and Health aid. Nursing provides medical and paramedical services that can be only provided by hospitals. On the other hand, health aid provides hygienic and essential daily care that elderly or disable people need. Basically, the main purpose of HHCS companies is to provide medical, paramedical, and social services for patients in their homes [3].

Due to the increase in the rate of care-dependent old people in Europe and adults with chronic disease in United States in last decade [2], the demand for HHC services is growing rapidly. Satisfying the highly expanding demand for the services requires companies to explore efficient planing and scheduling methods for delivering HHC staff to patients' home.

Motivated by the interest in developing effective planning means, many researchers all over the world have examined the problem from different perspectives and under various conditions. Begur et al. [4] and Cheng et al. [5] were among the first researchers who attempted to model and solve the scheduling and routing problem in HHC area. A spatial decision support system (SDSS) was suggested by Begur et al. [4] for a case study where the problem was solved by using scheduling heuristics. Cheng et al. [5] proposed two mathematical formulation and two phase heuristic solution approach for HHC problem by considering break time for service providers.

In general, the researches in the area of HHC can be categorized based on the planning horizon of the problem. In particular, we can define single-period and multi-period HHC problems where the former is related to the problems with planning horizon of one day and the later mentions to the problems considering multiple days as the scheduling horizon [6]. On the other hand, other researchers categorize the problem as short-term, medium-term, and long-term HHC problems. These categories refer to the problems with one day, one week, and more than one week as planning horizon, respectively [7].

The majority of the researchers deal with the scheduling and routing of HHC service providers by considering a single-period planning horizon. The single-period HHC problem can be investigated in different settings. Some researchers deal with the problem by focusing on *temporal precedence and synchronization* [8] [9] [10] and some others consider *work and break regulation* [7] [11] [7]. *Mode of transport* is another setting that can be different in the articles. Where most papers considers single mode for transporting the service providers to the patients' home, Hiermann et al. [12] and Rest et al. [13] incorporate public transport as another option. Also, Fikar et al. [14] assume that the health providers are delivered and picked up by a small bus fleet. In addition to the mentioned works which deal with the HHC problem in different settings, there are a few articles regarding *stochastic* HHC routing problem. In real-world cases, there is no perfect information before the operation. Therefore, to consider this situation, Lanzarone et al. [15] assumes HHC operations with stochastic demand whereas Yuan et al. [16] studies the problem under stochastic service time condition.

Single-period HHC problems are mainly modeled as extension of VRP in recent papers. Therefore, considering *travel time*, *travel distance*, and *cost of trips* as objective functions are very common in the literature. On the other hand, there are some works which consider the service provider's working time as the major cost. These studies try to include *overtime* and *waiting time* of the workers in the objective function [11] [12]. Another valid alternative for the objective of HHC is to maximize the number of served patients which is recently considered by Rasmussen et al. [17].

The constraints of short horizon HHC problems are mainly the same as VRPTW. *Demand satisfaction*, *time window*, *resource or capacity* restrictions are the common constraints that are considered in most of the papers. In these studies, all or most of the patients should be visited during their specified soft/hard time windows. In some problems, patients have specific time windows [11] [18] whereas in other types of problems time windows are assigned to each service provider [12] [19]. Besides, there are a few articles that considers *working time regulation* and *mandatory breaks* for service providers. Yuan et al. [16] sets a maximum total distance or duration for each service provider's route. Eveborn et al. [18] are among the pioneers who take into account *lunch breaks* for the nurses as *mandatory breaks*.

Multi-period HHC problems are different from single-period problems in terms of planning horizon. To adapt longer scheduling horizon, some changes should be applied to HHC problem. In the adapted version, all requests do not arise in the same time and the demand should be satisfied during a week or a month. Therefore, service providers should work for multiple days or weeks. These settings compel the problem to consider *stochastic demand* which can be arose randomly during the planning horizon. In this regard, Bennett et al. [20], Koeleman at al. [21] are among the researchers who deal with HHC problem in a dynamic setting and modeled the problem as a Markov decision process. Additionally, Carello et al. [22] and Rodriguez et al. [23] investigated the importance of stochastic demand and proposed solution methods based on the *stochastic programming* methods.

The objective function in the HHC problems with multi-period planning horizon, mainly, considers the service providing and staffing related factors. Although, there are some articles that focus on the travel related cost, the majority of the researchers emphasize on the providers' working time and continuity of care for the patients. Minimizing overtime cost, balancing the workload of service providers, and minimizing the number of care providers who visit the same patient (for considering continuity of care) are the most common objectives used in the literature. Besides, there are some works which deal with continuity of care in the constraints [22] [24]. Since, solving HHC problem in a dynamic and stochastic setting is more common in contrast to single-period horizon problem, some researchers deal with uncertainty by adding some constraints. For instance, adding chance constraints to the model or using stochastic programming techniques for dealing with uncertainty in the demand can be observed in the number of recent articles [20] [22] [23].

There are a number of recent works that developed a solution method for solving HHC problems. For solving single-period HHC problems majority of the researchers have developed meta-heuristics [12] [10] [13]. Mankowska et al. [10] implemented local search based procedure, whereas Hiermann et al. [12] compared the outputs of multiple metaheuristic on a single data. On the other hand, there are a few articles that implemented exact algorithm for solving HHC problems. Branch-and-price algorithm is predominant exact solution methodology which is implemented in recent studies for solving the problem optimally [19] [17] [16].

Majority of the solution methods proposed for multi-period HHC problem are exact solution procedures. Bard et al. [25] and Trautsamwieser et al. [7] developed a branch-and-price-and-cut algorithm to solve the problem optimally. Carello et al. [22] implemented robust optimization approach whereas Rodriguez et al. [23] applied stochastic programming technique to handle the uncertainty in demand and solve the problem optimally.

The novel idea that discriminates our problem from the problems in the literature is to have different types of service providers and group them to generate different service categories. Additionally, the patients are distinguished based on their required services and can be assigned to any appropriate service categories. This problem can be generalized into the cases where there are more than two types of service providers and we can define more service categories. Particularly, the problem is a combination of assignment and routing problem. The patients are assigned to a proper service category and then the routing problem of the vehicles which belong to the categories are solved. This idea was originally proposed by Tozlu et al. [26] who called the problem home care routing problem with time window and resource constraint (HCRPTWRC). Recently, they proposed a solution methodology based on Variable Neighborhood Search (VNS).

In this thesis, we deal with a single-period routing problem of HHCS personnel who provide two types of care services in a specific tight time windows that are predetermined by patients. In this problem, services are provided by a limited number of personnel (nurses and health care aids). Each patient requires either a nurse or a health care aid or both depending on their conditions. To the best of our knowledge, there is not any exact method proposed in the literature for solving the problem efficiently. Tozlu et al. [26] utilized CPLEX to exactly solve the mathematical formulation of the problem.

The main contribution of this thesis is an implementation of effective branch-and-price [27] algorithm to optimally solve HCRPTWRC in a novel setting. Branch-and-price has been proposed as an efficient methodology for solving variants of vehicle routing problems [28]. In general, Branch-and-Price algorithm is a beneficial tool for solving different types of Integer Programming (IP) and Mixed Integer Programming (MIP). The structure of the algorithm is as Branch-and-Bound algorithm which solves Linear Programming (LP) relaxation in each node of search tree. Branch-and-Price algorithm implements Column Generation algorithm in the nodes which solves the problem in a tighter feasible region and consequently, gives a better bound than LP relaxation.

For the implementation purpose, we formulate HCRPTWRC as set-partitioning problem. The proper structure of the problem enables us to decompose it into a master problem and three pricing sub-problems (one for each type of patients or vehicles). All of the sub-problems are elementary shortest path problems with time window. A dynamic programming approach called Label Setting Algorithm (LSA) [29] is implemented for solving the sub-problems. Using Branch-and-Price algorithm, the problem is optimally solved through column generation algorithm and embedded in a branch and bound structure. Numerical results are reported for selected instances generated based on Solomon's benchmarks.

The remainder of this thesis is organized as follows. Chapter 2 presents a mixed-integer programming formulation for HHC routing problem. Chapter 3 gives preliminaries for he branch-and-price algorithm that is implemented in this study and presents implementation issue regarding the method. Chapter 4 provides the results of our computational study and illustrates the efficacy of our branch-and-price algorithm. Chapter 5 concludes with final remarks and highlights some future research directions.

2 Problem Description, Formulation, and Solution method

In this section, first the problem definition will be presented. Then, mathematical formulation for the problem will be presented. At the end, an exact solution method that is implemented for solving the mathematical model will be explained.

2.1 Problem Description

In this problem, there is a set of patients each of whom require a certain type of service. Also, there are two types of personnel (i.e. two different resources), to provide the required services and the available number of each type of personnel is limited. Each patient has predetermined tight time window that the services should be provided within it. Vehicles, in this setting, carry at most two personnel. A feasible tour for a vehicle starts from the health center, visit some patients by considering their time windows, and return to the health center before completion time. In this problem, all the patients should be served within their tight time window and the resource limitation should also be considered.

In this problem, we assume that the demands and corresponding time windows are deterministic and we know them beforehand. We classify the patients based on the demanding services under three different types. Type 1 are the patients who require just nursing services, type 2 are the patients need just health aid services, and type 3 are the patients that ask both nursing and health aid services.

The major purpose of the HHCS provider company is to provide services for all of the patients in their specified time windows. Additionally, this company has to consider the limitation in the number of care personnel. During each day, the personnel are transported to patients home to provide services for them in a specific time window . There is no limitation on the number of vehicles, since, they are rented daily and paid per kilometer. Therefore, the objective of the company is to minimize the total distance traveled by the vehicles.

Current practice of the company is to solve the routing problem of the vehicles each transporting one nurse and one aid provider. This strategy might cause an inefficient utilization of the resources. Since there is a limited number of care provider staffs, with this strategy, some patients might not be visited during their specified time windows which leads to an infeasible problem.

In order to use the resources efficiently and overcome some infeasible cases, it has been proposed in [26] to define three types of vehicles. Vehicles type 1 each carries just one nurse, vehicles type 2 each transports one aid provider, and vehicles type 3 each conveys both a nurse and an aid provider. Note that in this strategy, vehicles type 1 and 2 can provide relevant services for patients type 1 and type 2, respectively. On the other hand, vehicles type 3 can serve all types of the patients. Additionally, we assume that a health aid provider cannot be substituted by a nurse and vice versa.

We provide a small example to demonstrate how this new practice may help the company. Suppose that the company has 2 nurses and 2 health aid provider staff and assume that the patients are as illustrated in Figure 2.1. In the case of current strategy that each vehicle caries one nurse and one aid provider, the company needs two vehicles and tries to solve the routing problem of two vehicles. From Figure 2.2, it can be seen that due to the time windows and the limitation in the number of HHCS staff, all patients cannot be visited once; therefore, this problem becomes infeasible. On the other hand, using the proposed strategy, as it can be observed from Figure 2.3, all patients can be visited in their predetermined time window at least once. In this case, we assure that we have at least one feasible solution for this problem, then, we can solve the optimization problem to find the best solution. Note that if the company can visit all customers with its current practice, this will also be one of the feasible solutions that can be obtained with our strategy. In this cases, the proposed strategy may also decrease the total traveled distance of feasible problems. Although the proposed strategy requires more vehicles in some cases, it makes the problem feasible. Since the fleet are leased and paid per kilometer, using more vehicle does not cause major problems for the company.

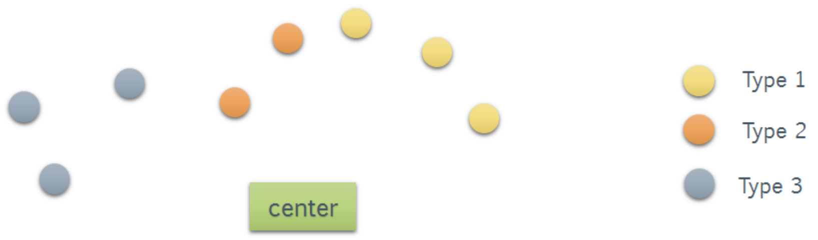

Figure 2.1: Illustration of a small example with different types of patients and a health center.

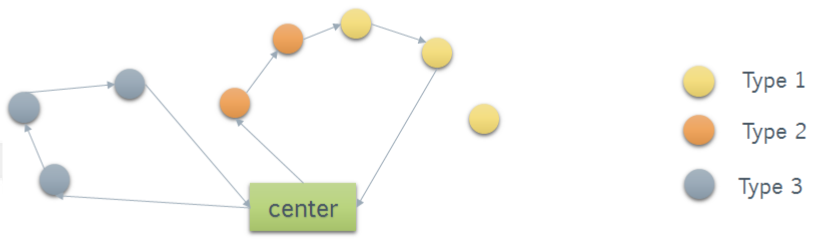

Figure 2.2: Best solution for the small example using current strategy of the company

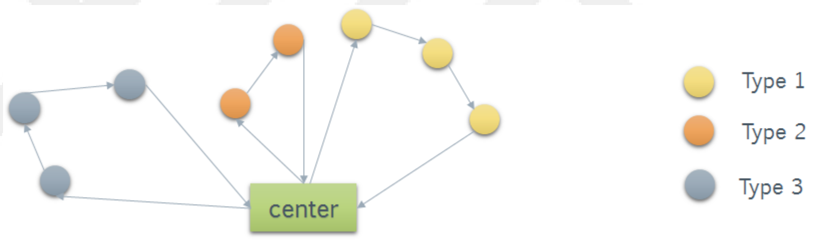

Figure 2.3: A feasible solution for the problem with proposed strategy.

HCRPTWRC in our setting is to minimize the total distance traveled by the vehicles, visiting all the patients in their specific time windows by considering resource limitations. In other words, the optimization problem is to assign each patient to an appropriate route by considering time windows and minimizing the total distance traveled by the compatible vehicles.

2.2 Mathematical Formulation

HCRPTWRC can be mathematically formulated as follows. Let N be the number of patients and $V = \{1, 2, \dots, N\}$ be the set of patients. Then, we define $V_0 = V \cup \{0\}$, $V_{N+1} = V \cup \{N+1\}$, and $V_{0,N+1} = V \cup \{0, N+1\}$ where 0 and $N+1$ are the health centers that each route should starts and ends, respectively. $G = (V_{0,N+1}, A)$ is a complete directed graph with $A = \{(i,j) | i, j \in V_{0,N+1}, i \neq j\}$, set of arcs. Each arc (i, j) has associated distance and travel time that we represent them by d_{ij} and t_{ij}, respectively. Additionally, $S = \{1, 2, 3\}$ contains the types defined for different patients based on their requested services and $R = \{1, 2\}$ represents the varied types for the resources (1 represents nurse, and 2 represents health aid). Also, let V^s be the set of patients of type $s \in S$ and h_r be the number of available staff type $r \in R$. Furthermore, in our setting, each patient i has a predetermined tight time window $[e_i, l_i]$ and a service time of st_i. Also, note that the time window of the health centers are $[0, T]$ which means that all tours should be completed before T.

Using the parameters that are defined in the previous paragraph, the mathematical formulation of HCRPTWRC can be written as following:

$$minimize \sum_{i \in V_{0,N+1}} \sum_{j \in V_{0,N+1}, j \neq i} \sum_{s \in S} d_{ij} x_{ij}^s \tag{1}$$

$$\text{subject to: } \sum_{i \in V_0, i \neq j} x_{i,j}^1 + \sum_{i \in V_0, i \neq j} x_{i,j}^3 = 1 \qquad \forall j \in V^1 \tag{2}$$

$$\sum_{i \in V_0, i \neq j} x_{i,j}^2 + \sum_{i \in V_0, i \neq j} x_{i,j}^3 = 1 \qquad \forall j \in V^2 \tag{3}$$

$$\sum_{i \in V_0, i \neq j} x_{i,j}^3 = 1 \qquad \forall j \in V^3 \tag{4}$$

$$\sum_{j \in V_{N+1}} x_{0j}^1 + \sum_{j \in V_{N+1}} x_{0j}^3 \leq h_1 \tag{5}$$

$$\sum_{j \in V_{N+1}} x_{0j}^2 + \sum_{j \in V_{N+1}} x_{0j}^3 \leq h_2 \tag{6}$$

$$\sum_{i \in V_0, i \neq j} x_{ij}^s = \sum_{i \in V_{N+1}, i \neq j} x_{ji}^s \qquad \forall j \in V, \forall s \in S \tag{7}$$

$$q_i + x_{ij}^s (t_{ij} + st_i) - T(1 - x_{ij}^s) \leq q_j \quad \forall i \in V_0, \forall j \in V_{N+1}, j \neq i, \forall s \in S \tag{8}$$

$$e_j \leq q_j \leq l_j \qquad \forall j \in V_{0,N+1} \tag{9}$$

$$x_{ij}^s \in \{0, 1\} \qquad \forall i \in V_0, \forall j \in V_{N+1}, j \neq i, \forall s \in S \tag{10}$$

$$q_j \geq 0 \qquad \forall j \in V_{0,N+1} \tag{11}$$

where x_{ij}^s is a binary decision variable indicating whether arc (i, j) is used by vehicle type s or not, also, the other decision variable is q_j which is the arrival time at patient j. The objective function (1) is to minimize the total distance traveled by the vehicles. Constraints $(2)-(4)$ ensure that all patients should be visited only once. Since patients type 1 can be served by vehicles type 1 and type 3, in writing constraint (2) we consider the vehicles of appropriate types. Analogous to type 1 patients, type 2 patients can be served by vehicles type 2 and type 3; therefore, constraint (3) takes into account the vehicles of both types. Constraint (4) just considers vehicles type 3, since patients type 3 can only be served by type 3 vehicles. Constraints (5) and (6) assure that the usage of the nurses and health aid providers is no more than the available staff. Since the problem is a variant of VRPTW, to assure the feasibility and elimination of the sub-tours we need constraints $(7)-(9)$ which are fundamental routing restrictions. Constraint (7) is a balance constraint and constraint (8) is a sub-tour elimination restriction in our problem. Moreover, constraint (9) makes all patients to be visited at their predetermined time windows. At the end, we have binary and non-negativity constraints.

Aforementioned formulation is a mixed integer programming (MIP) formulation of HCR-PTWRC. This formulation is originally introduced by Tozlu et al. (2015). To the best of our knowledge, any exact solution method for solving this problem has not been studied in the literature. In the remaining of this thesis we will introduce an exact solution method for solving HCRPTWRC to optimally. We implement the well known branch-and-price algorithm which is considered as one of the best solution methods for large MIPs [28].

3 Implementation Details

3.1 Branch-and-Price algorithm for Mixed Integer Programming Problems

In this section, we will introduce an efficient branch-and-price algorithm for HCRPTWRC. This algorithm is known as an effective solution method for solving large scale MIPs. In this regard, since HCRPTWRC is a variant of VRPTW, the exact solution methods that are proposed for solving VRPTW efficiently can be implemented for HCRPTWRC. In the following sub-section we will give a brief review on the structure of the branch-and-price algorithm.

3.1.1 Preliminaries

In this section, we will explain the fundamental components of a branch-and-price algorithm. In the literature, branch-and-price algorithm has been proposed to solve many variants of the VRP problem. Since, our problem can be modeled as a MIP, branch-and-price method is a promising solution methodology by an effective implementation.

The structure of the branch-and-price algorithm is analogous to branch-and-bound algorithm. The only difference is the solution method that is implemented in every node of the search tree. In fact, branch-and-price is an enhanced version of branch-and-bound algorithm. Let's look closely to the structure of branch-and-bound algorithm.

Given a mixed integer minimization program, the first step in implementing branch-and-bound algorithm is to drop integrality condition from the model and solve the LP-relaxation of the problem in the root node of the search tree. Since the feasible region of the original MIP is a subset of its LP-relaxation, the objective value of the LP-relaxation is a lower bound for the original minimization problem. The LP-relaxation can be solved by any linear programming (LP) solution methods i.e. simplex algorithm. In this step, if the optimal solution of the LP-relaxation is integer, then, this solution is also optimal for the original MIP and the algorithm will be terminated in the root node. However, if the optimal solution of the LP-relaxation is fractional, it will be required to perform branching and produce two or more child nodes [30]. Branching strategy is another important factor which affects the efficiency of the algorithm. In this regard, there are a lot of early researches which proposed problem dependent branching strategies (see [31] [32] [33]). The algorithm proceeds by choosing and solving a child node considering the branching strategy. Branch-and-bound algorithm terminates when no unprocessed child node exists. In the branch-and-bound algorithm a node can be fathomed because of three main reasons. First, if the problem in the node becomes infeasible, second, if the optimal solution obtained in that node is integer, and third, if the bound captured in the node is worst that the objective value of the best incumbent solution. These three rules help the algorithm to prune the branches and reduce the size of the search three. For more detail on the branch-and-bound algorithm see [30].

Early solution methods for discrete optimization were based on the explicit enumeration. These methods were not able to solve the problem with more than 30 integer decision variables [34]. For the large scale MIPs, since the number of potential solutions are exponentially many, the explicit enumeration has became impossible. In this regard, there are tools which help the algorithms to search the parts of the feasible region implicitly. Bounding the function that is optimized and using the best solution at hand assist the algorithms to perform efficiently [30]. Branch-and-bound algorithm utilizes some bounding techniques to efficiently search the feasible region. It uses the bounds that are obtained by solving the LP-relaxation of the problem and the objective function of the incumbent solution as lower and upper bounds.

It has been observed that the lower bound obtained from solving LP-relaxation is too weak and it causes the search tree to become immense. To overcome this problem, some techniques should be developed to generate strong bounds. Since the bounding is one of the important means to speed up the algorithm and increase the efficiency, many researchers have developed new bounding mechanisms. To the best of our knowledge, the first efficient bounding technique was introduced by Desrochers et al.(for more detail visit [31]) with the application in the VRPTW. They suggested to solve LP-relaxation of the set partitioning formulation of the VRPTW by column generation. The solution obtained by column generation provides an excellent lower bound that can be utilized in the branch-and-bound scheme.

Column generation is one of the efficient algorithms that was proposed for solving LPs in which the number of decision variables are exponentially many. In column generation algorithm the simplex steps that determines whether the current basic solution is optimal or not, and whether there is any non-basic variable with negative reduced cost or not are done by solving an optimization problem [35]. To the best of our knowledge, Ford and Fulkerson [36] were the first that proposed the idea of using column generation for solving LPs with the application in the multi-commodity maximum flow problem. The algorithm that they proposed can be generalized to solve any LPs. To this end, the mathematical model of problem should be written as set partitioning formulation. Additionally, the set of constraints should be partitioned into a set of master constraints and a set of sub-problem constraints. The resulting algorithm is Dantzing-Wolfe Decomposition [35].

After partitioning the problem into master problem and sub-problem, we begin column generation algorithm by solving a master problem that contains limited number of decision variables and we call it restricted master problem (RMP). Since, in large scale problems the number of decision variables are exponentially many, we do not consider all the variables in the beginning, we generate them as they are needed using the pricing sub-problems. In fact, objective function minimizing reduced cost along with the set of sub-problem constraints enable the pricing sub-problem to generate the column that corresponds to the coefficients of a non-basic decision variable in the set of master constraints. Additionally, the non-basic decision variable must have the most negative reduced cost to be a candidate variable for the basic solution of the RMP. Therefore, the optimal solution of the sub-problem must be a coefficient column corresponds to a non-basic decision variable which has the smallest reduced cost. If the reduced cost of the non-basic variable is non-negative, then, we can claim that the problem is optimal and there is no entering variable to the basic (optimality condition in Simplex algorithm). On the other hand, if the reduced cost is negative, then, the non-basic variable with corresponding coefficient column will be added to the RMP and it will be solved again. The iterations must be continued until sub-problem's optimal solution has non-negative reduced cost which means RMP has found the optimal solution.

Branch and Price algorithm is the combination of two aforementioned algorithms in which given a MIP, we implement branch-and-bound algorithm, and in every node of the search tree the LP-relaxation is solved using column generation. When the number of variables in the MIP problem is huge, using branch-and-price algorithm became crucial. There are interesting challenges in implementing column generation algorithm within the branch-and-bound framework. The most important one is the branching rules, since, the traditional branching strategies are not efficient and make the column generation more complicated. Researchers tried to find effective, but problem dependent, branching strategies to make the algorithm simple and more efficient. In this strategies, instead of branching on the variable of set partitioning problem, the branching is performed on a decision variable of the original problem. Which in this case making the master problem and sub-problem compatible with the branching strategy is more convenient. The early use of this branching strategy for a urban transit crew scheduling problem is give in [37] and for a pickup and delivery problem with time window is given in [38]. There are also other early studies which used this kind of branching strategy in the problems like graph coloring and airline crew scheduling (see [39] [40]).

In this thesis, we implement an efficient branch-and-price algorithm for solving HCRPTWRC. As the structure of the solution method, we implement a branch-and-bound algorithm and in every node of the search tree we perform column generation. Therefore, in every node of the search tree we need to define master problem and sub-problem based on the set partitioning formulation. In the following sub-sections we will introduced the set partitioning formulation of our specific problem.

3.1.2 Set partitioning formulation

In this section, we will introduce the set partitioning formulation of HCRPTWRC. For implementing column generation it is required to partition the set of constraints into a set of master constraints and a set of sub-problem constraints. For this purpose, from the original mathematical formulation of HCRPTWRC, we define constraints $(2) - (6)$ as a set of master constraints and $(7) - (11)$ as a set of sub-problem constraints. The objective of the sub-problem should calculate the reduced cost of basic and non-basic decision variables. As a result, master problem becomes a set covering problem which takes into account the number of staff utilized. On the other hand, sub-problem is a routing problem which tries to find a resource feasible shortest path from node 0 to node $N + 1$. Note that the sub-problem only considers the feasibility of the time windows, whereas, the master problem checks the feasibility of the number of staff transported.

In each iteration of the column generation algorithm, RMP is solved optimally, then, dual variables corresponds to the set of master constraints are obtained for the calculation of the reduced cost i.e. objective function of sub-problem. After updating the objective of the sub-problem, an elementary shortest path problem is solved optimally. If the objective value of the optimal solution is non-negative, then, the algorithm terminates and optimal solution of RMP is the optimal solution of the problem (LP-relaxation). On the other hand, if the objective value of the optimal elementary shortest path is negative, then, the coefficient column corresponding to this path is generated and added to the RMP. Afterwards, the RMP is solved again and the procedure is analogous.

Master problem. The mathematical formulation of HCRPTWRC as a set partitioning formulation can be obtained as following. If we define Ω^s as the set of routes of type $s \in S$ and c_p^s be the cost of path $p \in \Omega^s$ that is traveled by vehicle type $s \in S$, the master problem of set partitioning formulation can be written as following:

$$minimize \sum_{s \in S} \sum_{p \in \Omega^s} c_p^s \theta_p^s \tag{12}$$

$$\text{subject to: } \sum_{p \in \Omega^s} \alpha_{jp}^s \theta_p^s = 1 \qquad \forall s \in S, \forall j \in V^s \tag{13}$$

$$\sum_{s \in S} \sum_{p \in \Omega^s} \beta_{rp}^s \theta_p^s \leq h_r \qquad \forall r \in R \tag{14}$$

$$\theta_p^s \in \mathbb{N} \qquad \forall s \in S, \forall p \in \Omega^s \tag{15}$$

where θ_p^s is the number of times that route p of type s is used. The number of decision variables in the above formulation is exponentially many. Therefore, column generation is an efficient method for dealing with this problem. It can be comprehended from the formulation of the master problem that this problem is a set covering problem in which the aim is to visit all the patients only once by considering the resource constraints and minimizing the total cost. Since the problem is minimization problem and our cost matrix satisfies triangular inequality, we can replace constraint (13) with the following constraint and still get the solution which visits all the patients exactly once.

$$\sum_{p\in\Omega^s} \alpha_{jp}^s \theta_p^s \geq 1 \qquad \forall s \in S, \forall j \in V^s \tag{16}$$

by doing so, the dual variables corresponding to this set of constraints are restricted to be non-negative. This restriction causes the column generation algorithm to converge faster [28]. In implementing the algorithm, since the number of decision variables are huge, we start with restricted master problem which contains only subset of decision variables. We will generate the non-basic variables and add them to the RMP as they are requires. As we have discussed earlier, this procedure is done by sub-problem. In the following of this section, we will introduce definition of the column and the sub-problem of the set partitioning formulation.

Column Definition. Definition of a column corresponding to the optimal solution of the sub-problem should be compatible with the structure of RMP. We can define the column as following:

$$\alpha_p^s = \begin{bmatrix} \alpha_{1p}^s \\ \alpha_{2p}^s \\ \vdots \\ \alpha_{jp}^s \\ \vdots \\ \alpha_{Np}^s \\ \beta_{1p}^s \\ \beta_{2p}^s \end{bmatrix} \tag{17}$$

where α_{jp}^s indicates whether vehicle type s visits patient j through route p or not which can be formulated as following:

$$\alpha_{jp}^s = \sum_{i \in V_0, i \neq j} x_{ijp}^s \qquad \forall s \in S, \forall j \in V^s, \forall p \in \Omega^s \tag{18}$$

where x_{ijp}^s is the decision variable of the original mathematical formulation. If path p visits patient j, then, the value of α_{jp}^s is 1 and 0 otherwise. Also, parameters β_{ip}^s are 1 if the vehicle type s through route p carries resource type i and 0 otherwise.

For illustration, suppose there are 5 patients, ($N = 5$) and $V_{0,N+1} = \{0, 1, 2, \ldots, N+1\}$. Also, assume that based on the required services the patients can be categorized as following:

$V^1 = \{1, 4\}$ (patients type 1), $V^2 = \{2, 3\}$ (patients type 2), and $V^3 = \{5\}$ (patients type 3).

Lets assume that in one of the iterations of the column generation algorithm the optimal shortest path is to start from the health center 0, then visit 2 and 3, then go to the health center $N + 1$. In this case, the path p visits only patients of type 2. Therefore, α_{2p}^2 and α_{3p}^2 are equal to 1 whereas other α_{jp}^s are equal to 0.

Additionally, as we discussed earlier, patients type 1 ask for nursing service, patients type 2 require health aid services, and patients type 3 need both of them. Since the path p is a type 2 vehicle that visits only type 2 patients it carries a health aid provider which means that

β^2_{1p} equals to 0 whereas β^2_{2p} equals to 1. As a result, the column corresponding to the path p which will be added to the RMP looks like following:

$$\alpha^2_p = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{19}$$

Similarly, for the other paths based on the type of vehicle and the patients that were visited we can write the corresponding coefficient column and add the column to the RMP. Next we will define the set partitioning formulation of HCRPTWRC master problem.

Pricing sub-problem. The pricing sub-problem of HCRPTWRC should be able to produce a non-basic resource feasible route with the smallest reduced cost. Therefore, for obtaining the objective function of the sub-problem we need to use the original formulation of the reduced cost from Simplex algorithm. Lets Ω be the set of all routes, then, the reduced cost can be identified as following:

$$\overline{c}_p = c_p - \overline{y}a_p \qquad \forall p \in \Omega \tag{20}$$

where c_p is the cost of path p, $\overline{y}$ is a vector corresponding to the dual variables of the master problem, and a_p is the column that will be generated. In our setting, this column is the same as the one which is represented by (17). For writing the objective function of the sub-problem, we need to define dual variables of our master problem. Let γ^s_j and ω_r be the dual variables corresponding to the constraints (13) and (14), respectively. Using the original formulation of reduced cost and the the notation of our problem, the reduced cost of a path p

of type s can be calculated as following:

$$\overline{c}_p^s = \sum_{(i,j)\in A^s} d_{ij}x_{ij}^s - \sum_{j\in V^s} \gamma_j^s \alpha_{jp}^s - \sum_{r\in R} \omega_r \beta_{rp}^s \qquad \forall s \in S, \forall p \in \Omega^s \tag{21}$$

using the definition of α_{jp}^s in (17) and simplifying the equation, we can modify the equation above and obtain the following equality:

$$\overline{c}_p^s = \sum_{(i,j)\in A^s} (d_{ij} - \gamma_j^s)x_{ij}^s - \sum_{r\in R} \omega_r \beta_{rp}^s \qquad \forall s \in S, \forall p \in \Omega^s \tag{22}$$

The objective function of the sub-problem can be considered as the minimization of the expressions in the right hand side of the equation (22). The decision variable, x_{ij}^s, only exists in the first expression. The second expression is a parameter and we call it *dual constant*. For the sake of simplicity, we drop dual constant from the objective function of the sub-problem, since, it does not affect the optimal solution of the sub-problem. When the optimal solution of the sub-problem is obtained, we use dual constant to check the optimality condition (whether reduced cost is negative or not). Eventually, the objective function of the sub-problem looks like following:

$$minimize \sum_{s\in S} \sum_{(i,j)\in A^s} \overline{c}_{ij}^s x_{ij}^s \tag{23}$$

where $\overline{c}_{ij}^s = d_{ij} - \gamma_j^s$, is called the modified cost of an arc (i,j) in the directed graph of our problem.

Consequently, the sub-problem of set partitioning formulation of HCRPTWRC can be written as following:

minimize $\sum_{s\in S} \sum_{(i,j)\in A^s} \overline{c}_{ij}^s x_{ij}^s$

subject to: (7)-(11)

Let's look at the structure of the sub-problem closely. Since, all the constrains are written for each type of vehicles $s \in S$, also, the objective is minimizing the summation over all vehicle types, we can state that the structure of the sub-problem is decomposible. The pricing sub-problem can be decomposed into $|S|$ numbers of independent sub-problems. In fact, decomposition helps us to partition the feasible region of the sub-problem into smaller regions which accelerates the solution algorithms. From algorithmic point of view, it enables us to solve the sub-problems in parallel and each time instead of sending one column with negative reduced cost to the RMP, we could send one column per sub-problem with negative reduced cost (if there exists any). This parallelization provides more convenience and less computational time. Therefore, the mathematical formulation for sub-problem(s) can be written as following:

sub-problem(s):

$$minimize \sum_{(i,j)\in A^s} \overline{\mathrm{c}}^s_{ij} x^s_{ij} \tag{24}$$

$$\text{subject to: } \sum_{i\in Q^s_0, i\neq j} x^s_{ij} = \sum_{i\in Q^s_{N+1}, i\neq j} x^s_{ji} \qquad \forall j \in Q^s \tag{25}$$

$$q_i + x^s_{ij}(t_{ij} + st_i) - T(1 - x^s_{ij}) \leq q_j \qquad \forall i \in Q^s_0, \forall j \in Q^s_{N+1}, j \neq i \tag{26}$$

$$e_j \leq q_j \leq l_j \qquad \forall j \in Q^s_{0,N+1} \tag{27}$$

$$x^s_{ij} \in \{0,1\} \qquad \forall i \in Q^s_0, \forall j \in Q^s_{N+1}, j \neq i \tag{28}$$

$$q_j \geq 0 \qquad \forall j \in Q^s_{0,N+1} \tag{29}$$

where Q^s is the set of patients that can be served by vehicle type $s \in S$. In our problem, since we have three different types of vehicles, our sub-problem is decomposed into three sub-problems. From the mathematical formulation $(24) - (29)$, it can be observed that every pricing sub-problem is an *elementary shortest path* problem. There are different methods to solve this problem in the literature. Note that vehicle type 1 and type 2 can serve the patients type 1 and type 2, respectively; therefore, we can state that $Q^s = V^s$ for $s = 1, 2$. However, this is not true for vehicles type 3. Since these vehicles can serve all types of the patients, the corresponding set Q contains all the patients ($Q^3 = V$).

Up to now, we gave a general information regarding branch-and-price algorithm and introduced the set partitioning formulation of HCRPTWRC. In the following section, we will discuss the implementation issue and the techniques that we utilized to improve the performance of the branch-and-price algorithm.

3.2 Implementation Issues

We develop branch-and-price algorithm to solve HCRPTWRC. There are some factors which affect the efficiency of the algorithm. In this section, we will touch upon these factors and how we implement different parts of the algorithm for the problem.

3.2.1 Initial Solution

Initial solution is one of the crucial factors which affects the convergence of the column generation algorithm in each node of the search tree. In the column generation algorithm, at the first step, we need to solve RMP which contains only a subset of decision variables. These variables should be able to create an initial basic feasible solution. Therefore, it is important to choose or create the initial variables such that the initial RMP becomes feasible. There are different ways to generate initial variables and their corresponding coefficient columns.

One of the common ways to generate initial columns is to use artificial columns. This idea comes from *phase I* of the Simplex algorithm. In phase I, we usually add artificial decision variables to the mathematical formulation and assign a very large number (in minimization case) as their costs. Since, they are variables with high cost, Simplex algorithm in each iteration tries to find better solution and get rid of the artificial variables. In fact, the problem is feasible if in the optimal solution there is no artificial variables. If there exists any artificial variable in the basic optimal solution, then, the problem is infeasible.

The procedure in the column generation algorithm is analogous to Simplex method. In column generation algorithm, the artificial columns are able to create initial feasible solution; however, these columns might not be a feasible column for the original problem. Therefore, by assigning a very high cost to these columns, we can get rid of them in the next iterations of the column generation algorithm.

In our setting, we tried to generate artificial columns in two ways. First, we generated a column corresponding to a route which visits all customers twice. Also, this route does not use any resources. We call this initial solution **Artificial Single**. The column of the initial solution ensures the existence of the initial basic feasible solution. However, it is not feasible in context of original problems, since, it does not consume any resources. We assign a large number as its cost in the objective function of the RMP. Therefore, column generation algorithm, in each iteration, tries to find a good route to throw the artificial column out from the basic solutions. For illustration purpose, assume we have 3 patients and 2 types of resources. Our initial artificial column looks like following:

$$\alpha_{initial} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 0 \\ 0 \end{bmatrix} \tag{30}$$

where the number of elements which equal to 2 is the same as the number of patients, and elements equal to 0 indicates that the corresponding route does not consume any resources.

The second method that we have considered for generating the artificial columns, is to use the columns of the identity matrix as initial routes. Each column of the identity matrix corresponds to a route which starts from the health center, visits a particular patient and returns to the health center. These routes are feasible for the original problem. However, if we assign resource consumption to these columns, the number of resources required will be more than the number of available staff and the columns will not be able to generate initial feasible solution. Therefore, to avoid the infeasibility of the RMP, we do not assign resource consumption to the initial routes. The resulting initial solution is what we call **Artificial Identity**. For the illustration purpose, let's again assume that there are 3 patients and 2 different types of resources, the initial columns that we use is as following:

$$A_{initial} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{31}$$

where each column of the initial A matrix corresponds to a coefficient column of an artificial decision variable in the RMP. In this method, the initial matrix contains N artificial column.

These two methods each has positive and negative effects on the performance of the branch-and-price algorithm. To analyze these effects, the algorithm is run in two settings and the results are presented in Section 4. We report how the efficiency of the branch-and-price algorithm is affected by implementing these two methods.

3.2.2 Solving the sub-problem

In this thesis, we considered four approaches for dealing with pricing sub-problem in each iteration of column generation algorithm. First, we used CPLEX 12.6.2 for solving the sub-problem optimally. Since, this method is computationally inefficient, then, we implemented two enhanced variants of *Label Correcting Algorithm* for solving pricing problem which are proposed by Feillet et al. [29] and Boland et al. [41]. Finally, we perform a heuristic pricing, in which we limit the size of label sets.

Solving the sub-problem might be the most crucial part in the implementation of branch-and-price algorithm. Since, it affects the performance of the algorithm directly, using efficient techniques for solving the sub-problems becomes more important. To this end, there are many methods proposed in the literature for solving the sub-problems. The simplest way to solve the sub-problem of the column generation algorithm is to use the off-the-shelf solvers. These solvers assist researchers to solve the sub-problem as LP or MIP in a straightforward way. However, using these solvers at every pricing iteration is computationally expensive. Therefore, researchers began to look closely to the structure of the pricing sub-problems and tried to develop problem dependent solution methods.

The structure of some MIPs are such that the corresponding pricing sub-problem looks like a well-known problem. We can mention to Cutting Stock problem which is very famous in column generation area. Consider that we write the mathematical formulation of the Cutting Stock problem as a set partitioning problem and decompose the problem into a master problem and a sub-problem. In this case, the pricing sub-problem is a well-known Knapsack problem [42]. Thus, if we apply column generation algorithm for solving Cutting Stock problem, at pricing iterations we can take the advantage of efficient algorithms which were proposed for solving Knapsack problem.

In our problem which is a variant of VRPTW, the pricing sub-problem is an *Elementary Shortest Path* problem [43]. Therefore, we can benefit from the algorithms suggested for solving this specific problem. One of the algorithms in this area is *Label Correcting Algorithm* which is proposed by Desrochers [44] and is suitable for solving *shortest path problem with resource constraints*. However, this algorithm is not appropriate for the *Elementary Shortest Path* problems in which exsistance of cycles are not allowed. In this regards, Feillet [29] adapted Desrochers' algorithm to solve the *elementary shortest path problem with resource constraints* by adding binary node-visit resources to the definition of the labels. Node-visit resource of a particular node in a label is consumed (equals to 1) if the partial path corresponding to the label has visited that node.

Label correcting algorithm is a dynamic programming approach that tries to find an optimal path from source node s to sink node t by considering the resource limitations. In this algorithm, every label represents a partial path and contains elements which correspond to cost and resource consumption of the partial path. Additionally, for adapting label correcting algorithm for elementary shortest path problems, binary node-visit resources must be considered by the labels. This algorithm starts with a trivial label (with all elements equal to 0) from the source node and tries to extend this label along an arc if the extension is feasible. The extension is infeasible, if the resulting partial path is not resource-feasible or if it cannot reach to the sink node because of the resource limitations.

The performance of the label correcting algorithm depends mostly on the dominance rule that eliminates partial paths. Let P_i' and P_i^* be two distinct partial paths from source to node i. Also, let (C', R') and (C^*, R^*) be the corresponding labels, where, C' and C^* are costs, and R' and R^* are resource consumption of P_i' and P_i^*, respectively. Path P_i' dominates path P_i^* if and only if $(C', R') \neq (C^*, R^*)$, $C' \leq C^*$, and $R_k' \leq R_k^*$ for $\forall k \in K$ where $|K|$ is the number of resources.

To strengthen the dominance relation and consequently eliminate more partial paths, Feillet et al. [29] introduced the concept of *unreachable nodes* and proposed a new definition for the labels. For a path P_i from the source node to a node i, a node v is said to be *unreachable* if it has already been visited by P_i, or it cannot be visited because of the resource limitation. Using this notation, the description of the labels can be modified. In the new definition, node-visit resource of a particular node in a label is consumed when the node is *unreachable* for the partial path corresponding to the label.

In this thesis, we define a label, extension of a label, and the dominance relation as following.

Definition of a label. A label corresponding to a partial path from source node 0 to node i can be defined as $\Lambda_i = (\theta, \tau, \Gamma, P, \rho)$, where, θ represents the reduced cost of the partial path, τ shows the beginning time of the service at node i, $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_N\}$ is a visit-vector representing availability of node-visit resource of each node, P tracks the sequence of the nodes in the partial path, and ρ is the cost of the route.

Label extension. Extension of a label $\Lambda_i = (\theta, \tau, \Gamma, P, \rho)$ from node i to node j creates label $\Lambda_j = (\theta', \tau', \Gamma', P', \rho')$, where:

- $\theta' = \theta + \overline{c}_{ij}$
- $\tau' = max\{e_j, \tau + st_i + t_{ij}\}$
- $\Gamma' = \{\gamma_k = 1$ if k is an *unreachable* node for label Λ_j; 0 otherwise$\}$
- $P' = P \cup \{j\}$
- $\rho' = \rho + c_{ij}$

Note that τ' which represents the starting time of the service at node j, cannot occur before the specified time window e_j. Therefore, if a service provider reach the patient earlier that its time window, s/he should wait to start service at e_j. Furthermore, definition of an *unreachable* node is the same as proposed description. Specifically, node k is an *unreachable* node for label Λ_j if:

- node k has already been visited via partial path corresponding to label Λ_j,
- Λ_j cannot extend to node k because of the time restriction.

- extension of the resulting label, Λ_k, to the health center exceeds the completion time, T.

Dominance relation. Let $\Lambda_i = (\theta, \tau, \Gamma, P, \rho)$ and $\Lambda'_i = (\theta', \tau', \Gamma', P', \rho')$ be two label on node i. Label Λ_i dominates Λ'_i if:

- $\theta \leq \theta'$
- $\tau \leq \tau'$
- $\gamma_r \leq \gamma'_r, \forall r \in V$

There is another variation of *label setting algorithm* which is introduced by Boland et al. [41] and in called *general label setting algorithm* (GLSA). Boland et al. [41] enhanced the algorithm proposed by Feillet et al. [29] by defining a set of *unprocessed labels* which keeps track of the labels on a node that are not extended to another label yet. Also, for accelerating Feillet's algorithm, they proposed *state space augmenting* algorithm called *GLSA(S)* in which they are trying to find a lower bound on the *elementary shortest path* problem by defining node-visit resources only for the nodes contained in set S.

In this work, we implement the *label setting algorithm* proposed by Feillet et al. [29] and *GLSA* suggested by Boland et al. [41] which is slightly modified version of the previous one. In each iteration of *label setting algorithm* the later algorithm suggests to just extend the labels included in the set of *unprocessed labels* and do not extend all the labels in each iteration. This version enhances the performance of the algorithm, specially, when the number of labels on each node is huge.

Solving pricing sub-problems by *label correcting algorithm* enhances the efficiency of the branch-and-price algorithm. However, in some instances in which every patient has a wide time window, the performance of the *label correcting algorithm* is not excellent. In these instances, the number of feasible partial paths grows rapidly which implies that the number of labels on each patient is huge. Therefore, the algorithm slows down and dynamic programming approach seems to be an inefficient approach to deal with the pricing problem of these instances.

As the forth approach, we implement a variant of *label correcting algorithm* in which we limit the number of feasible labels on every patient. This method, accelerates the column generation algorithm and consequently enhances the performance of the branch-and-price algorithm. In every pricing problem, we first try to find a feasible path with a negative reduced cost using this approach. Then, if the proposed method fails to find such a path, the original *label correcting algorithm* which we described in the previous paragraphs will be invoked.

The four approaches that were discussed in this section are implemented and the performance of the branch-and-price algorithm under each approach is reported in Section 4.2.

3.2.3 Branching Scheme

Branching strategy is another crucial point of the branch-and-price algorithm. When optimal solution of the *restricted master problem* is fractional, we need to perform branching to achieve an integer solution. There are several strategies for branching i.e. branching on fractional variable of master problem (θ_p^s), variable of the original formulation (x_{ij}^s), and number of vehicles. Among those techniques, branching on the variables of the original formulation is more common and admitted to be more convenient [43].

In branching on the variables of the original formulation, we need to select an arc (i, j) with a fractional flow, $0 < f_{ij} < 1$. Then, we obtain two branches: in one branch we forbid arc (i, j) to be a part of the solution ($f_{ij} = 0$), in another branch we enforce arc (i, j) to be a part of the solution ($f_{ij} = 1$). The flow of an arc can be calculated using solution values of the master problem. It is computed by summing values of those decision variables which correspond a path containing arc (i, j). Our strategy for selecting an arc is to choose arc (i, j) which its flow is closest to 0.5.

Once branching performed, two child nodes are generated and in each child node the column generation algorithm should be performed compatible with the branching decision. This requires modifications in the master problem and the sub-problems of the child nodes. To modify master problem, in each child node, we simply discard all the columns from master problem and rebuild it with an artificial initial solution. On the other hand, making sub problem compatible with the branching decisions is somehow tricky.

To enforce an arc (i, j) to be a part of solution, we discard all the arcs (i, l) with $l \neq j$ and (l, j) with $i \neq l$ from the pricing sub-problems. This modification makes the sub-problems generate paths which visit node j only after node i. Also, consider that in the master problem we have the restriction of visiting all nodes at least one time. The modification along with the restriction enforce arc (i, j) to be a part of the optimal solution. On the other hand, to forbid an arc (i, j) to be a part of solution, we simple discard this arc from the sub-problems. Note that since we have three sub-problems we need to modify all of them in every node of the search tree. For more detail on branching scheme see [43].

4 Computational Results

We implement a branch-and-price algorithm to solve HCRPTWRC. In this section we will discuss the performance of the algorithm and the effect of different strategies that we have used.

4.1 Instance Generation

In the computational experiment of this thesis, we utilized modified version of the instances generated by Tozlu et al. [26]. The instances are adapted version of the benchmark Solomon instances which have 25 and 50 nodes. For instances with 25 patients, C101, C106, C207, C208, R103, R108, R201, R210, RC101, RC105, RC201, and RC205 are chosen and the information regarding coordinates and the time windows are obtained. To adapt these instance to the HCRPTWRC, it is required to assign each patients a service type and service time.

In HCRPTWRC, three different types of services are defined. Therefore, in the adapted versions, the instances can be grouped based on the percentage of the patients requiring each type of services. The groups are called G1, G2 and G3 which are generated based on the information presented in Table 4.1. For example, 60%, 48%, and 48% of the patients in G1, G2, and G3, respectively, require type 1 services.

Table 4.1: Percentage of each service type in different groups

	Service type		
Groups	1	2	3
G1	60%	32%	8%
G2	48%	36%	16%
G3	48%	28%	24%

Based on the data in Table 4.1, three variations of the Solomon's instances are generated; thus, 36 adapted instances with 25 patients are generated for our problem. The service time of each patient is related to the type of the service that s/he requested. In fact, service time of type 1, type 2, and type 3 patients are 10, 40, and 45 minutes, respectively.

Other important data that should be determined are the number of available nurses and health aid providers. This part is not a simple task, since, the problem can become infeasible if the number of resources are too tight. On the other hand, if the resources are too loose, then, resource constraints will become redundant.

Based on the tightness and looseness of the resource parameters, 4 different variants of the 36 instances can be produced. The resource level in these 4 variants can be considered as tight-tight, loose-tight, tight-loose, and loose-loose. For obtaining loose and tight levels of the resources, HCRPTWRC $((1)-(11))$ solved optimally using different objective functions. The optimal solution obtained by solving the problem using minimization of the resources as the objective function is considered as tight resources. On the other hand, loose resources are achieved by solving the problem with the objective of distance minimization when resource constraints are relaxed. Finally, there are generated 144 varied instances containing 25 patients.

Similarly, for the instances with 50 patients, we had the same strategy. However, in defining the resource levels, since, CPLEX is not able to solve the instances with 50 patients, we used the reported optimal number of vehicles for each of the Solomon's instances. Therefore, we create 36 instances containing 50 patients.

In this work, HCRPTWRC is considered as a single-period routing problem. Also, it is assumed that the service providers should complete their route within 4 hours. This assumption is valid since most of the HHCS providers are a hospital or a health center personnel. Therefore, they initiate the routes after their part time duty in the hospital. This working schedule is convenient and preferable. Since, for long time window routing problems the *working time regulations* and *mandatory breaks* must be considered.

To adapt the special setting to the problem we need to select appropriate instances. Among the mentioned instanced, $R103$, $R08$, $RC101$, and $RC105$ are compatible with the settings of our problem. Therefore, for implementing the branch-and-price algorithm, we select appropriate variants of these groups from instance pool.

The characteristics of the instances are demonstrated in Table 4.2. First column shows the groups that each instance belongs to them. They are determined by using the information of Table 4.1. Second and third columns are names and IDs of the instances. Following column is optimal objective function value of the instances (OFV). Next two columns refer to the available number of nurses and health aid providers for each instances. Based on the available resources, the service levels are determined in the next column; L_L, T_T, and medium mention to the loose, tight, and medium level resources. Finally, the last column is the completion time (CT) of the problem in which all vehicles visiting the patients should return to the health center.

For generating the instances with 50 patients we follow the same instructions. In this case, it is a hard task to generate feasible instances. As initial resources, we used the optimal vehicle numbers that are reported by Solomon. This gives us insights about the approximate number of required staff. However, in some cases, these resource levels are inadequate and we need to increase the number of available nurses and health aid providers. Also, we are not sure that whether specific increase in the resource levels will make the instance feasible or not. First, we solve the instances with the optimal vehicle numbers as resource levels. Then, if the problem is feasible, we decrease the resource levels to generate feasible medium and tight instances. On the other hand, if the instance with the optimal vehicle numbers as resources is infeasible, then, we assume that there are large numbers of available staff and solve the generated instances with CPLEX. After catching feasible instances, we decrease the resource levels to generate feasible medium and tight instances.

Among the generated instances, we pick the feasible ones to implement our branch-and-price algorithm. Table 4.3 represents the characteristics of the instances utilized. Since the number of feasible instances are limited, in addition to Solomon's *R103*, *R108* instances we used *RC201*, and *RC205* instances which have longer completion time. Note that *RC101* and *RC105* have become infeasible; therefore, we did not consider them. In Table 4.3, the first column illustrates the groups that each instance belongs to them. These groups are determined based on the percentage of the Type 3 patients. Next two columns are the names and the IDs of the instances. The number of available nurses and health aid providers are men-

Table 4.2: Characteristics of the instances with 25 patients

				Resources			
Group	Name	ID	OFV	#N	# aid	Level	CT(s)
G1	R103wd25-model-08-1-1	1	501	7	6	L_L	230
	R103wd25-model-08-1-2	2	556	5	4	T_T	230
	R103wd25-model-08-1-3	3	536	5	5	medium	230
	R103wd25-model-08-1-4	4	531	6	4	medium	230
G2	R103wd25-model-16-1-1	5	553	7	7	L_L	230
	R103wd25-model-16-1-2	6	607	5	6	T_T	230
	R103wd25-model-16-1-3	7	604	5	7	medium	230
	R103wd25-model-16-1-4	8	585	6	6	medium	230
G3	R103wd25-model-24-1-1	9	569	7	7	L_L	230
	R103wd25-model-24-1-2	10	605	6	5	T_T	230
	R103wd25-model-24-1-3	11	580	6	6	medium	230
	R103wd25-model-24-1-4	12	601	7	5	medium	230
G1	R108wd25-model-08-1-1	13	443	5	5	L_L	230
	R108wd25-model-08-1-2	14	456	4	4	T_T	230
	R108wd25-model-08-1-3	15	451	4	5	medium	230
	R108wd25-model-08-1-4	16	456	5	4	medium	230
G2	R108wd25-model-16-1-1	17	479	6	6	L_L	230
	R108wd25-model-16-1-2	18	480	5	6	T_T	230
	R108wd25-model-16-1-3	19	480	5	7	medium	230
	R108wd25-model-16-1-4	20	479	7	6	medium	230
G3	R108wd25-model-24-1-1	21	470	6	6	L_L	230
	R108wd25-model-24-1-2	22	490	5	6	T_T	230
	R108wd25-model-24-1-4	23	470	7	6	medium	230
G1	RC101wd25-model-08-1-1	24	598	7	7	L_L	240
	RC101wd25-model-08-1-2	25	684	4	5	T_T	240
	RC101wd25-model-08-1-3	26	630	4	6	medium	240
	RC101wd25-model-08-1-4	27	669	5	5	medium	240
G2	RC101wd25-model-16-1-1	28	672	7	8	L_L	240
	RC101wd25-model-16-1-2	29	796	4	7	T_T	240
	RC101wd25-model-16-1-3	30	695	4	8	medium	240
	RC101wd25-model-16-1-4	31	694	5	7	medium	240
G3	RC101wd25-model-24-1-1	32	663	8	7	L_L	240
	RC101wd25-model-24-1-2	33	699	5	7	T_T	240
	RC101wd25-model-24-1-3	34	715	6	6	medium	240
	RC101wd25-model-24-1-4	35	671	6	7	medium	240
G1	RC105wd25-model-08-1-1	36	602	7	7	L_L	240
	RC105wd25-model-08-1-2	37	648	4	6	T_T	240
	RC105wd25-model-08-1-3	38	633	4	7	medium	240
	RC105wd25-model-08-1-4	39	630	5	6	medium	240
G2	RC105wd25-model-16-1-2	40	679	5	5	T_T	240
	RC105wd25-model-16-1-3	41	660	5	6	medium	240
	RC105wd25-model-16-1-4	42	667	6	5	medium	240
G3	RC105wd25-model-24-1-2	43	701	5	6	T_T	240
	RC105wd25-model-24-1-3	44	701	5	7	medium	240
	RC105wd25-model-24-1-4	45	687	6	6	medium	240

OFV, objective function value; #N, number of nurses; #aid, number of health aid providers; L_L, loose-loose; T_T, tight-tight; CT, completion time.

Table 4.3: Characteristics of the instances with 50 patients

Group	Name	ID	#N	# aid	CT(s)
G1	R103wd50-model-08-1-1	1	10	10	230
G1	R103wd50-model-08-1-3	2	9	8	230
G1	R108wd50-model-08-1-1	3	9	9	230
G1	R108wd50-model-08-1-2	4	8	8	230
G2	R108wd50-model-16-1-1	5	10	10	230
G3	R108wd50-model-24-1-1	6	10	10	230
G1	RC201wd50-model-08-1-1	7	4	4	960
G2	RC201wd50-model-16-1-1	8	4	4	960
G3	RC201wd50-model-24-1-1	9	3	4	960
G1	RC205wd50-model-08-1-1	10	5	5	960
G2	RC205wd50-model-16-1-1	11	5	5	960
G3	RC205wd50-model-24-1-1	12	5	5	960

#N, number of nurses; #aid, number of health aid providers; CT, completion time.

tioned in the next two columns. Finally, the completion time of each instance is demonstrated in the last column.

In the following section, we represent and compare the results obtained from various implementations of the branch-and-price algorithm on the generated instances.

4.2 Results and Discussion

In this section, we will evaluate the performance of the branch-and-price algorithm and the benefit of several techniques which were introduced. The algorithm is implemented in Java using the branch-and-price framework of Java OR Library (Jorlib). For solving *restricted master problem* CPLEX 12.6.2 is utilized. All computational experimets are performed on 64-bit server with Intel Xeon E5-2640 v3 processor with a speed of 2.6 GHz and 128 GB RAM. The time limit for instances including 25 is 1 hour and instances with 50 patients is 4 hours.

To evaluate the effect of different techniques that were introduced for solving pricing subproblems, we conduct several experiments. In each experiment we initiate our branch-and-price algorithm with different pricing solvers. The pricing solvers are the implementations of the *label setting algorithm* which we discussed in Section 3.2.2. We refer to *label setting algorithm* proposed by Feillet et al.[29] as *GLSA*, and the algorithm suggested by Boland et al.[41] as *Boland_GLSA*. Additionally, the heuristic approach in which we have limited the number of labels on every node is denoted by *GLSA_L*.

For reporting purpose, we record the CPU time in which the problem is solved, Pricing time that is the total duration that pricing sub-problems are solved, number of columns generated and number of iterations of the column generation algorithm. Table 4.4, Table 4.5, and Table 4.6 represent the computational results for the branch-and-price algorithm with different pricing solvers. We categorize Solomon's modified instances based on their service levels, then the reported outcomes are calculated by averaging over the outcomes in each category.

The computational results for the branch-and-price algorithm with *GLSA* pricing solver is presented in Table 4.4. To enhance the performance of the algorithm, we utilize the aggregation of *GLSA* and *GLSA_L* as a pricing solver. Table 4.5 show the results for the later pricing solver. It can be perceived from the tables that *GLSA_L* assists the *GLSA* to solve pricing sub-problems more efficient. It can be seen that *GLSA_L* enhances the solution time, pricing time, number of columns generated, and iterations of the column generation for all instances of *R103*, tight instances of *R108*, and loose instances of *RC105*. Also, we can observe that *GLSA_L* can boost the branch-and-price algorithm to solve the medium instances of *R108*, medium and tight instances of *RC105* in a shorter time while it increases the number of columns and iterations. On the other hand, while *GLSA_L* enhances the pricing solver for some of the instances, it does not affect the *RC101* instances. Comparing the results of both solvers, number of iterations and columns remain same but by applying heuristic method the solution time and pricing time are increased. Which means that heuristic pricing cannot produce any column with negative reduced cost. This implies that the columns with negative reduced cost are deleted from the heuristic approach by limiting number of labels on every node of a network. Additionally, *GLSA_L* cannot enhance the performance of the pricing solver in the loose instances of *R108*.

By examining the tables in this section, it can be seen that *Boland_GLSA* + *GLSA_L* solver

Table 4.4: Computational results for BAP with GLSA solver

ID	R_L	CPU Time(s)	Pricing Time(s)	# colmns	# iterations
R103	L_L (3)	8.35	8.05	2056.67	1081.00
	medium (6)	19.31	1219.89	4385.50	2153.00
	T_T (3)	5.24	14.82	3794.67	1672.33
R108	L_L (3)	55.82	55.28	3113.33	1757.33
	medium (5)	57.22	56.78	2670.20	1509.00
	T_T (3)	888.89	879.29	59971.00	33753.67
RC101	L_L (3)	2.93	1.55	3676.67	2140.00
	medium (6)	0.52	0.44	395.33	210.83
	T_T (3)	0.63	0.52	507.67	293.33
RC105	L_L (1)	13.58	12.38	8307.00	5063.00
	medium (6)	170.19	155.20	115103.17	63657.67
	T_T (3)	6.69	6.25	2783.00	1441.00

CPU Time, the duration that problem is solved; Pricing Time, the duration that pricing sub-problems are solved; #columns, total number of columns generated; #iterations, total number of iterations of column generation.

Table 4.5: Computational results for BAP with GLSA+GLSA_L solver

ID	R_L	CPU Time(s)	Pricing Time(s)	# colmns	# iterations
R103	L_L (3)	3.02	2.82	1111.00	588.33
	medium (6)	6.16	5.80	2283.50	1193.67
	T_T (3)	4.41	4.15	1609.33	785.67
R108	L_L (3)	65.94	62.66	19364.33	11420.00
	medium (5)	22.65	21.65	5579.80	3272.40
	T_T (3)	126.61	122.08	27980.67	16634.00
RC101	L_L (3)	3.13	2.57	3676.67	2140.00
	medium (6)	0.55	0.47	395.33	210.83
	T_T (3)	0.71	0.61	507.67	293.33
RC105	L_L (1)	7.75	6.96	5830.00	3613.00
	medium (6)	131.14	116.73	116889.67	64699.67
	T_T (3)	4.62	4.18	3201.00	1656.00

CPU Time, the duration that problem is solved; Pricing Time, the duration that pricing sub-problems are solved; #columns, total number of columns generated; #iterations, total number of iterations of column generation.

outperforms *GLSA + GLSA_L* and *GLSA* solvers. In all of the instances, *Boland_GLSA + GLSA_L* solver enhances the CPU time and pricing time. In some instances this enhancement is really huge, whereas in some others it is negligible.

Table 4.6: Computational results for BAP with Boland_GLSA+GLSA_L solver

ID	R_L	CPU Time(s)	Pricing Time(s)	# colmns	# iterations
R103	L_L (3)	2.75	2.56	1111.00	588.33
	medium (6)	4.78	4.44	2284.17	1194.17
	T_T (3)	3.32	3.08	1609.33	785.67
R108	L_L (3)	51.09	48.16	19364.67	11420.33
	medium (5)	16.01	15.16	5579.80	3272.40
	T_T (3)	71.22	66.81	28070.00	16689.33
RC101	L_L (3)	2.91	2.40	3676.67	2140.00
	medium (6)	0.52	0.45	395.33	210.83
	T_T (3)	0.68	0.59	507.67	293.33
RC105	L_L (1)	7.46	6.74	5830.00	3613.00
	medium (6)	115.93	103.04	117075.50	64799.00
	T_T (3)	4.10	3.67	3201.00	1656.00

CPU Time, the duration that problem is solved; Pricing Time, the duration that pricing subproblems are solved; #columns, total number of columns generated; #iterations, total number of iterations of column generation.

In addition to the analyzing the effects of different pricing solvers, we examine the efficacy of different initial solutions. To this end, we used two distinguished initial solutions that are discussed in Section 3.2.1. To evaluate the performance of the branch-and-price algorithm in two settings, we solve the instances including 25 patients using *GLSA* as pricing solver. The detail results of using different initial solutions can be found in Section 6.

From the results, we observe that in general Artificial Identity outperforms Artificial Single in terms of solution time, number of iterations of column generation, number of columns generated through column generation, and the number of nodes of search tree. Particularly, Artificial Identity solves 77.77% of the instances faster and 88.88% of them with less iterations. Also, it outdoes in 86.66% of the instances by producing less columns during the column generation algorithm. In terms of the number of nodes in the search tree, in 48.88% of the instances the results are same for both of the artificial initial solutions. Artificial Identity and Artificial Single surpasses the other one in 31.11% and 20.00% of the instances, respectively.

Finally, to be able to evaluate the performance of the branch-and-price algorithm, we solve the instances using CPLEX 12.6.2. The best incumbent solution, the best lower bound on the objective function value, CPU time, and integrality gap are recorded for CPLEX. Then, we implement the enhanced version of the branch-and-price (BAP) algorithm, in which the initial solution is an artificial identity matrix described in Section 3.2.1; also, Boland's *GLSA* along with heuristic pricing, mentioned in Section 3.2.2, solves the pricing sub-problems in every iteration of the column generation algorithm.

For reporting the results, the instances are categorized based on their service levels and the outcomes are averaged in each category. Table 4.7 presents the result for the instances with 25 patients which are solved within 1 hour. In the table, computational time and integrality gap, if the instance is not solved within the time limit, are presented for CPLEX and the enhanced branch-and-price (BAP); also, the number of nodes in the search tree are represented for BAP.

From Table 4.7, it can be observed that BAP solves all the instances optimally, while CPLEX is not able to solve most of the instances in R103 and R108 categories. Particularly, the performance of BAP exceeds CPLEX in solving the instances with medium and tight resource levels. Although the average CPU time of BAP for RC101 type instances is slightly more than that of CPLEX, it can be perceived BAP solves R103, R108, and RC105 more efficient than CPLEX. The detail information regarding the computational time and optimal objective function value can be found in Section 6.

Note that, for the instances which are not solved within the time limit, the BAP Gap is calculated a ratio $(z^*_{IP} - z^*_{LP}) \setminus z^*_{IP}$, where z^*_{IP} is objective value of the best incumbent solution and z^*_{LP} is the objective value of the LP-relaxation solve by column generation in the root node. Also, CPLEX Gap is the same as the gap reported by CPLEX.

In addition to the instances including 25 patients, we solve the instances including 50 patients. Table 4.8 illustrates the results of the enhanced branch-and-price (BAP) algorithm and CPLEX. We set 4 hours limit to the solvers to solve the instances. From the table, it

Table 4.7: Comparing the outcomes of BAP and CPLEX for the instances with 25 patients

		CPLEX		BAP		
ID	R_L	Time(s)	GAP(%)	Time(s)	# N	Gap(%)
R103	L_L (3)	80.52	0.00	2.75	9.67	0.00
	medium (6)	1869.42	3.19	4.78	19.00	0.00
	T_T (3)	3603.08	8.19	3.32	13.00	0.00
Average		1855.61	3.64	3.91	15.17	0.00
R108	L_L (3)	216.52	0.00	51.09	164.33	0.00
	medium (5)	1086.99	3.21	16.01	48.20	0.00
	T_T (3)	2578.69	5.95	71.22	209.00	0.00
Average		1256.42	3.08	40.63	123.73	0.00
RC101	L_L (3)	0.76	0.00	2.91	45.67	0.00
	medium (6)	0.56	0.00	0.52	4.67	0.00
	T_T (3)	0.31	0.00	0.68	6.33	0.00
Average		0.54	0.00	1.16	15.33	0.00
RC105	L_L (1)	54.24	0.00	7.46	69.00	0.00
	medium (6)	129.35	0.00	115.93	1521.67	0.00
	T_T (3)	69.21	0.00	4.10	33.00	0.00
Average		825.17	0.00	71.54	929.80	0.00

R_L, Resource level; BAP Time, branch-and-price CPU time; # N, number of nodes in the search tree of BAP.

can be observe that BAP is able to solve 6 of the instances optimally, while CPLEX solves 4 instances among them to optimally. In two instances that CPLEX could not solve them, BAP outperforms and solves them in a short duration. Additionally, there are 3 instances that CPLEX is not able to solve them within the time limit (we mark them with dashes). The BAP algorithm outperforms in these instances and could find an incumbent solution for each of them. Also, there are instances (*1*,*2*,*3*) that the best solution found by BAP surpass the ones found by CPLEX. Which also means that the integrality gap of BAP is less than the that of CPLEX.

To recapitulate, in this section, we evaluate the performance of the proposed branch-and-price algorithm (BAP) by comparing its outcomes with the results of the CPLEX. It has been observed that in most of the cases performance of BAP excels CPLEX. Then we examined the efficacy of proposed techniques for boosting the branch-and-price algorithm. To this end, we implement the algorithm using three different algorithms as pricing solvers; *GLSA*, *GLSA*+ *GLSA_L*, and *Boland_GLSA* + *GLSA_L*. Among the solvers, in most of the cases *Boland_GLSA* + *GLSA_L* surpass the others in terms solution time, number of columns and iterations of column generation algorithm. Second, we investigate the effects of initial solutions. For this purpose, we implement the branch-and-price algorithm using two different artificial initial solutions; Artificial Identity, and Artificial Single. In this case, the performance of the algorithm is better when we use Artificial Identity columns.

Table 4.8: Comparing the outcomes of BAP and CPLEX for the instances with 50 patients

	CPLEX				BAP					
File ID	Best Solution	LB	Time(s)	GAP(%)	CG-Bound	Best Solution	LB	Time(s)	Gap(%)	# N
1	919	668.47	14590.00	27.26	881.85	912	912.00	14283.24	3.31	4873
2	964	689.36	14404.07	28.49	881.15	937	881.85	14254.89	5.96	7654
3	817	515.02	14436.91	36.96	724.07	764	764.00	14361.83	5.23	1140
4	-	–	–	–	728.22	1112	728.22	14359.02	34.51	1496
5	-	–	–	–	772.83	1226	1226.00	14343.79	36.96	2789
6	-	–	–	–	822.16	837	837.00	14356.21	1.77	1420
7	785	785.00	77.75	0	785.00	785	785.00	17.27	0.00	1
8	801	801.00	87.75	0	798.55	801	801.00	91.28	0.00	9
9	997	997.00	1362.51	0	982.17	997	997.00	2646.26	0.00	145
10	627	588.71	14407.13	6.11	627.00	627	627.00	15.03	0.00	1
11	628	628.00	5235.07	0	628.00	628	628.00	18.66	0.00	1
12	621	603.80	14401.75	2.77	621.00	621	621.00	16.43	0.00	1

Best Solution, objective of the best incumbent solution; Time, the duration that the problem is solved; GAP, optimality gap; # N, number of nodes in the search tree of BAP.

5 Conclusions and Future Research Directions

Dealing with home health care routing problems is one of the most innovative ways to reduce the transportation costs and raise the satisfaction of the patient by serving them on time. In real word examples, the structure of the problem is complicated. Therefore, finding novel ways to simplify and model the problem in order to make them tractable is so valuable. Using effective methods to solve the problems exactly is another crucial aspect that assist service provider companies to systematically use the resource and serve the patients. To this end, we developed an efficient branch-and-price algorithm to solve the problems up to 50 patients.

In this thesis, we examined the efficacy of different techniques for improving the algorithm. We observe that solving the pricing sub-problem efficiently is one of the factors that affects the overall performance of the branch-and-price algorithm. Also, the results illustrates that initial solution is another key to boost the algorithm. Therefore, implementing more efficient pricing solver and utilizing more appropriate initial solutions will be the considerations of our future work.

In current work, we assume deterministic demand, travel time, service time, and time windows. They are strong assumptions and in real cases there might be stochastic demands and corresponding time windows. Also, the travel times of a part of a route might depends on the congestion hours. Therefore, stochastic demand and travel times are valid considerations which can be another opportunity for the future works.

Furthermore, this problem can be generalized to the cases where more that two types of service providers exist. In this case, the major consideration will be how we can group the service providers and patients in order to cover all of the demands and minimize/maximize the considered objective function. This problem is a bit challenging. However, a novel idea and mathematical formulation can simplify the problem. Proposing a general mathematical formulation for handling this problem and exact or heuristic solution methods afterwards, can provide new opportunities for companies to enhance their service providing systems.

References

[1] B. Issaoui, I. Zidi, E. Marcon, F. Laforest, and K. Ghedira, "Literature review: Home health care," in *2015 15th International Conference on Intelligent Systems Design and Applications (ISDA)*, pp. 485–492, Dec 2015.

[2] D. S. Mankowska, F. Meisel, and C. Bierwirth, "The home health care routing and scheduling problem with interdependent services," *Health care management science*, vol. 17, no. 1, pp. 15–30, 2014.

[3] S. Yalçındag, A. Matta, and E. Sahin, "Human resource scheduling and routing problem in home health care context: a literature review," *ORAHS/Cardiff, United Kingdom*, 2011.

[4] S. V. Begur, D. M. Miller, and J. R. Weaver, "An integrated spatial dss for scheduling and routing home-health-care nurses," *Interfaces*, vol. 27, no. 4, pp. 35–48, 1997.

[5] E. Cheng and J. L. Rich, "A home health care routing and scheduling problem," *Houston, Texas*, 1998.

[6] C. Fikar and P. Hirsch, "Home health care routing and scheduling: A review," *Computers & Operations Research*, vol. 77, pp. 86–95, 2017.

[7] A. Trautsamwieser and P. Hirsch, "A branch-price-and-cut approach for solving the medium-term home health care planning problem," *Networks*, vol. 64, no. 3, pp. 143–159, 2014.

[8] D. Bredström and M. Rönnqvist, “Combined vehicle routing and scheduling with temporal precedence and synchronization constraints,” *European journal of operational research*, vol. 191, no. 1, pp. 19–31, 2008.

[9] R. B. Bachouch, A. Guinet, and S. Hajri-Gabouj, “A decision-making tool for home health care nurses planning,” in *Supply Chain Forum: an International Journal*, vol. 12, pp. 14–20, Taylor & Francis, 2011.

[10] D. S. Mankowska, F. Meisel, and C. Bierwirth, “The home health care routing and scheduling problem with interdependent services,” *Health care management science*, vol. 17, no. 1, pp. 15–30, 2014.

[11] A. Trautsamwieser, M. Gronalt, and P. Hirsch, “Securing home health care in times of natural disasters,” *OR spectrum*, vol. 33, no. 3, pp. 787–813, 2011.

[12] G. Hiermann, M. Prandtstetter, A. Rendl, J. Puchinger, and G. R. Raidl, “Metaheuristics for solving a multimodal home-healthcare scheduling problem,” *Central European Journal of Operations Research*, vol. 23, pp. 89–113, Mar 2015.

[13] K.-D. Rest and P. Hirsch, “Daily scheduling of home health care services using time-dependent public transport,” *Flexible Services and Manufacturing Journal*, vol. 28, no. 3, pp. 495–525, 2016.

[14] C. Fikar and P. Hirsch, “A matheuristic for routing real-world home service transport systems facilitating walking,” *Journal of Cleaner Production*, vol. 105, pp. 300–310, 2015.

[15] E. Lanzarone and A. Matta, “Robust nurse-to-patient assignment in home care services to minimize overtimes under continuity of care,” *Operations Research for Health Care*, vol. 3, no. 2, pp. 48–58, 2014.

[16] B. Yuan, R. Liu, and Z. Jiang, “A branch-and-price algorithm for the home health care scheduling and routing problem with stochastic service times and skill requirements,” *International Journal of Production Research*, vol. 53, no. 24, pp. 7450–7464, 2015.

[17] M. S. Rasmussen, T. Justesen, A. Dohn, and J. Larsen, “The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies,” *European Journal of Operational Research*, vol. 219, no. 3, pp. 598–610, 2012.

[18] P. Eveborn, P. Flisberg, and M. Rönnqvist, “Laps carean operational system for staff planning of home care,” *European Journal of Operational Research*, vol. 171, no. 3, pp. 962–976, 2006.

[19] A. Dohn, E. Kolind, and J. Clausen, “The manpower allocation problem with time windows and job-teaming constraints: A branch-and-price approach,” *Computers & Operations Research*, vol. 36, no. 4, pp. 1145–1157, 2009.

[20] A. R. Bennett and A. L. Erera, “Dynamic periodic fixed appointment scheduling for home health,” *IIE Transactions on Healthcare Systems Engineering*, vol. 1, no. 1, pp. 6–19, 2011.

[21] P. M. Koeleman, S. Bhulai, and M. van Meersbergen, “Optimal patient and personnel scheduling policies for care-at-home service facilities,” *European Journal of Operational Research*, vol. 219, no. 3, pp. 557–563, 2012.

[22] G. Carello and E. Lanzarone, “A cardinality-constrained robust model for the assignment problem in home care services,” *European Journal of Operational Research*, vol. 236, no. 2, pp. 748–762, 2014.

[23] C. Rodriguez, T. Garaix, X. Xie, and V. Augusto, “Staff dimensioning in homecare services with uncertain demands,” *International Journal of Production Research*, vol. 53, no. 24, pp. 7396–7410, 2015.

[24] P. M. Duque, M. Castro, K. Sörensen, and P. Goos, “Home care service planning. the case of landelijke thuiszorg,” *European Journal of Operational Research*, vol. 243, no. 1, pp. 292–301, 2015.

[25] J. F. Bard, Y. Shao, X. Qi, and A. I. Jarrah, “The traveling therapist scheduling problem,” *IIE Transactions*, vol. 46, no. 7, pp. 683–706, 2014.

[26] B. Tozlu, R. Daldal, T. nlyurt, and B. atay, "Crew constrained home care routing problem with time windows," in *2015 IEEE Symposium Series on Computational Intelligence*, pp. 1751–1757, Dec 2015.

[27] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations Research*, vol. 46, no. 3, pp. 316–329, 1998.

[28] G. Ozbaygin, O. E. Karasan, M. Savelsbergh, and H. Yaman, "A branch-and-price algorithm for the vehicle routing problem with roaming delivery locations," *Transportation Research Part B: Methodological*, vol. 100, pp. 115–137, 2017.

[29] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen, "An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems," *Networks*, vol. 44, no. 3, pp. 216–229, 2004.

[30] J. Clausen, "Branch and bound algorithms-principles and examples," *Department of Computer Science, University of Copenhagen*, pp. 1–30, 1999.

[31] M. Desrochers, J. Desrosiers, and M. Solomon, "A new optimization algorithm for the vehicle routing problem with time windows," *Operations research*, vol. 40, no. 2, pp. 342–354, 1992.

[32] M. Savelsbergh, "A branch-and-price algorithm for the generalized assignment problem," *Operations research*, vol. 45, no. 6, pp. 831–841, 1997.

[33] R. Anbil, R. Tanga, and E. L. Johnson, "A global approach to crew-pairing optimization," *IBM Systems Journal*, vol. 31, no. 1, pp. 71–78, 1992.

[34] L. Wolsey, *Integer Programming*. Wiley Series in Discrete Mathematics and Optimization, Wiley, 1998.

[35] G. L. Nemhauser, "Column generation for linear and integer programming," *Optimization Stories*, vol. 20, p. 64, 2012.

[36] L. R. Ford Jr and D. R. Fulkerson, “A suggested computation for maximal multi-commodity network flows,” *Management Science*, vol. 5, no. 1, pp. 97–101, 1958.

[37] M. Desrochers and F. Soumis, “A column generation approach to the urban transit crew scheduling problem,” *Transportation Science*, vol. 23, no. 1, pp. 1–13, 1989.

[38] Y. Dumas, J. Desrosiers, and F. Soumis, “The pickup and delivery problem with time windows,” *European journal of operational research*, vol. 54, no. 1, pp. 7–22, 1991.

[39] R. Anbil, R. Tanga, and E. Johnson, “A global optimization approach to crew scheduling,” *IBM Systems Journal*, vol. 31, pp. 71–78, 1991.

[40] A. Mehrotra and M. A. Trick, “A column generation approach for graph coloring,” *informs Journal on Computing*, vol. 8, no. 4, pp. 344–354, 1996.

[41] N. Boland, J. Dethridge, and I. Dumitrescu, “Accelerated label setting algorithms for the elementary resource constrained shortest path problem,” *Operations Research Letters*, vol. 34, no. 1, pp. 58–68, 2006.

[42] A. Prékopa and C. I. Fábián, *Cutting-stock problem Cutting-Stock Problem*, pp. 594–595. Boston, MA: Springer US, 2009.

[43] D. Feillet, “A tutorial on column generation and branch-and-price for vehicle routing problems,” *4OR*, vol. 8, pp. 407–424, Dec 2010.

[44] M. Desrochers, *An algorithm for the shortest path problem with resource constraints*. École des hautes études commerciales, Groupe d’études et de recherche en analyse des décisions, 1988.

6 APPENDIX A: RESULTS OF ALL INSTANCES FOR DIFFERENT BAP IMPLEMENTATIONS

In this section we present the results for each of the instances in detail. Table 6.1 and 6.2 include the OFV*, optimal objective function value, and Time, duration that instances are solved, for CPLEX and the enhanced branch-and-price (BAP) algorithm. The performance of the branch-and-price algorithm in two different settings where the initial solutions are different are also demonstrated. Table 6.3 and Table 6.4 provide the results for every instance while implementing Artificial Identity and Artificial Single as initial solutions. The tables give detail information about the solution time, number of iterations and columns of column generation, and number of nodes in the search tree.

Table 6.1: The results of BAP and CPLEX for instances including 25 patients

	CPLEX		**BAP**	
ID	OFV*	Time(s)	OFV*	Time(s)
1	501	34.913	501	5.04
2	556	3602.61	556	7.15
3	536	1445.71	536	8.8
4	531	613.396	531	2.7
5	553	72.774	553	0.776
6	612	3603.94	607	2.18
7	604	3601.94	604	4.667
8	585	1110.18	585	6.2
9	569	133.88	569	2.43
10	605	3602.69	605	0.615
11	580	845.151	580	4.18
12	605	3600.16	601	2.16
13	443	137.733	443	6.74
14	456	3604.26	456	1.45
15	451	442.497	451	8.36
16	474	3602.91	456	1.35
17	479	401.968	479	0.91
18	480	528.063	480	0.96
19	480	648.48	480	0.96
20	479	385.026	479	1.00

Table 6.2: The results of BAP and CPLEX for instances including 25 patients (cont'd)

	CPLEX		**BAP**	
ID	OFV*	Time(s)	OFV*	Time(s)
21	470	109.856	470	145.61
22	490	3603.76	490	211.25
23	470	356.041	470	68.37
24	598	1.341	598	3.1
25	684	0.406	684	0.21
26	630	1.232	630	0.2
27	669	0.421	669	0.17
28	672	0.468	672	3.44
29	796	0.094	796	0.45
30	695	0.421	695	0.14
31	694	0.515	694	1.5
32	663	0.468	663	2.19
33	699	0.421	699	1.38
34	715	0.296	715	0.64
35	671	0.453	671	0.48
36	602	54.242	602	7.46
37	648	160.416	648	4.21
38	633	144.348	633	2.68
39	630	165.47	630	25.66
40	679	23.634	679	3.75
41	660	101.666	660	78.05
42	667	26.333	667	20.21
43	701	23.588	701	4.34
44	701	317.789	701	401.91
45	687	20.498	687	167.09

Table 6.3: The results of the branch-and-price algorithm with Artificial Identity initial solution and GLSA pricing solver

ID	Time(s)	# iterations	# columns	# nodes
1	1.58	1049	2045	21
2	21.15	1295	3046	43
3	42.14	4002	8386	99
4	5.33	283	604	7
5	1.06	54	92	1
6	4.95	264	577	7
7	13.15	686	1411	17
8	39.37	3205	6440	85
9	3.16	245	431	5
10	1.11	32	75	1
11	9.1	563	1100	13
12	4.04	159	357	5
13	44.61	389	762	7
14	10.94	56	111	1
15	119.15	707	1436	11
16	9.7	45	100	1
17	2.81	67	116	1
18	2.82	59	107	1
19	2.92	59	107	1
20	2.75	67	116	1
21	335.72	8311	14519	139
22	2339.62	62670	110652	1147
23	328.56	8312	14517	139
24	4.68	3209	5206	81
25	0.2	32	63	1
26	0.23	38	71	1
27	0.19	30	62	1
28	1.46	850	1396	27
29	0.35	89	185	3
30	0.15	33	61	1
31	1.18	596	1077	17
32	2.06	1289	1960	33
33	1.09	521	925	15
34	0.52	162	317	5
35	0.39	115	190	3
36	22.92	7815	12030	185
37	4.99	1126	2114	33
38	3.03	537	938	13
39	45.06	11991	21935	323
40	4.64	673	1501	27
41	102.11	31220	62370	1101
42	19.94	4734	10309	183
43	5.79	1144	2155	31
44	285.22	114235	206751	3907
45	246.8	93304	180843	3715

Table 6.4: Results of the branch-and-price algorithm with Artificial Single initial solution and GLSA pricing solver

ID	Time(s)	# iterations	# columns	# nodes
1	17.12	2095	4085	35
2	46.89	4644	10601	117
3	43.74	4717	9843	89
4	15.26	1021	2189	19
5	0.95	67	121	1
6	5.46	370	771	7
7	16.3	1092	2147	19
8	13.37	1322	2561	25
9	10.8	1126	2045	21
10	1.23	46	106	1
11	8.8	721	1450	13
12	4.16	237	512	5
13	52.81	818	1593	11
14	9	63	139	1
15	125.92	969	1936	13
16	6.13	71	140	1
17	2.86	74	133	1
18	2.87	75	134	1
19	3.1	75	134	1
20	2.85	74	133	1
21	139.83	4769	8395	67
22	3332.97	101159	179725	1461
23	217.21	6432	11147	99
24	3.1	2288	4043	45
25	0.31	49	106	1
26	0.37	63	115	1
27	0.32	52	107	1
28	3.43	2688	4638	61
29	0.39	140	300	3
30	0.27	57	107	1
31	1.44	813	1580	17
32	2.13	1485	2425	31
33	1.6	755	1238	15
34	0.64	240	427	5
35	0.85	376	626	7
36	13.17	5074	8326	103
37	7.14	1755	3300	33
38	3.19	724	1265	13
39	48.41	15502	28312	329
40	5.4	1006	2113	23
41	136.71	62361	118039	1491
42	45.44	14166	28385	347
43	6.96	1615	3.35	31
44	518.83	182344	320588	4239
45	241.4	113490	207115	2825