MODELING, ANALYSIS, AND EXACT ALGORITHMS FOR SOME BIOMASS LOGISTICS SUPPLY CHAIN DESIGN AND ROUTING PROBLEMS

Maichel Miguel Aguayo Bustos

Dissertation submitted to the Faculty of

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Industrial and Systems Engineering

Subhash C. Sarin, Chair

Douglas R. Bish

John S. Cundiff

Barbara M. P. Fraticelli

June 29, 2016

Blacksburg, Virginia

Keywords: Biomass Logistics, Routing Problems, Harvesting Scheduling, Asymmetric Traveling salesman Problem, Vehicle Routing Problems (VRP).

MODELING, ANALYSIS, AND EXACT ALGORITHMS FOR SOME BIOMASS LOGISTICS SUPPLY CHAIN DESIGN AND ROUTING PROBLEMS

Maichel Miguel Aguayo Bustos

(ABSTRACT)

This dissertation focuses on supply chain design and logistics problems with emphasis on biomass logistics and routing problems. In biomass logistics, we have studied problems arising in a switchgrass-based bio-ethanol supply chain encountered in the Southeast, and a corn stover harvest scheduling problem faced in the Midwest Unites States, both pertaining to the production of cellulosic ethanol. The main contributions of our work have been in introducing new problems to the literature that lie at the interface of the lot-sizing and routing problems, and in developing effective exact algorithms for their solution.

In the routing area, we have addressed extensions of the well-known traveling salesman and vehicle routing problems. We have proposed new formulations and have developed exact algorithms for the single and multiple asymmetric traveling salesmen problems (ATSP and mATP), the high-multiplicity asymmetric traveling salesman problem (HMATSP) and its extensions, and the fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB). Furthermore, we have introduced a new strategy to reduce routing cost in the classical vehicle routing problem (VRP).

MODELING, ANALYSIS, AND EXACT ALGORITHMS FOR SOME BIOMASS LOGISTICS SUPPLY CHAIN DESIGN AND ROUTING PROBLEMS

Maichel Miguel Aguayo Bustos

(GENERAL AUDIENCE ABSTRACT)

This dissertation focuses on the modeling, analysis, and develop of exact algorithms for some biomass logistics supply chain design and routing problems. In particular, we study a biomass logistics supply chain design problem (BL-SCDP), a dynamic corn stover harvesting scheduling problem (CSHSP), the well-known asymmetric traveling salesman problem (ATSP) and multiple asymmetric traveling salesman (mATSP), the high-multiplicity asymmetric traveling salesman problem (HMATSP) and its extensions, the fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB), and the vehicle routing problem with temporary storage (VRP-TS).

The first problem that we have investigated in this dissertation is a Biomass Logistic Supply Chain Design Problem (BL-SCDP) that arises in a switchgrass-based bio-ethanol supply chain, and lies at the interface of the single-item capacitated lot-sizing problem with multiple parallel facilities and time-dependent selective high-multiplicity multiple traveling salesmen problem. The problem is formulated as a mixed-integer program, and a branch-and-price-based method is developed for its solution that relies on effective implementation strategies. Our computational investigation reveals efficacy of the proposed method over direct solution of its model formulation by CPLEX 12.5.1 in obtaining near-optimal solutions for large-sized problem instances that are inspired by a real-life case study.

The second problem that we have studied in this work is a corn stover harvesting scheduling problem in a dynamic environment (CSHSP). A cellulosic ethanol plant contracts with farmers to harvest corn stover after the harvest of grain has been completed. Farmers call in to the plant to schedule the harvest from their fields. The plant has a fleet of harvesting crews under contract, which must be assigned to harvest the fields as they are "called in" over the harvest season, assumed to be split into T periods. The static CSHSP, in which the call-in times of the fields are assumed known at the beginning of the harvest season, can be modeled as a time-dependent heterogeneous high-multiplicity multiple traveling salesmen problem with given time windows. These time windows require that the fields to be harvested between their ready times (call-in times) and due dates (the date by which fields must be harvested based on the contract signed with the farmers). A mixed-integer programming formulation is developed to model this real-world optimization problem, and a periodic re-optimization scheme is proposed to solve the dynamic CSHSP, in which the static CSHSP is solved repeatedly at different decision points throughout the harvest season in a rolling horizon manner. The effectiveness of the proposed approach is demonstrated by solving a real-life harvest season in the Midwest United States.

The third problem that we have addressed is the well-known asymmetric traveling salesman problem (ATSP) and the multiple asymmetric traveling salesmen problem (mATSP). We present an algorithm to solve the ATSP and mATSP by generating violated subtour elimination constraints (SECs) from specific integer solutions. Computational results for the ATSP reveal that the proposed approach is able to solve 29 out of 33 well-known instances taken from the literature (involving between 100 and 1001 cities) to optimality within one hour of CPU time, and it outperforms the direct solution by CPLEX of the three most effective existing formulations for this problem. Furthermore, the proposed approach is demonstrated to be competitive with the most effective state-of-the-art exact algorithms available for the ATSP in the literature. For the mATSP, the proposed approach is able to solve 27 out of 36 instances derived from the ATSP library (involving up to 1001 cities)

to optimality within one hour of CPU time, and it also outperforms the direct solution by CPLEX of the three most effective formulations reported in the literature for this class of problems. The effectiveness of the proposed method is further exhibited by its ability to solve all mATSP instances of size greater than 443 cities with a maximum optimality gap of only 2.18%, while only one instance of size greater than 443 cities can been solved by one of the existing formulations available in the literature with an optimality gap of 17.71%. Given this efficacy of the proposed approach, it might be useful to the researchers and practitioners in the context of applying ATSP either as a stand-alone model or as a sub-model within some application setting.

The fourth problem that we have investigated is the high-multiplicity asymmetric traveling salesman problem (HMATSP), which is an extension of the traveling salesman problem where a city can be visited multiple times. We show that even though this formulation is not as tight as the best known formulation for the HMATSP, it is faster and easier to use for direct solution by CPLEX as well as in a Benders decomposition framework. We also exhibit the versatility of the proposed formulation by modeling several variants or extensions of the HMATSP that have not been studied in the literature. Furthermore, we propose effective accelerated Benders algorithms to solve instances of the HMATSP and its extensions that are derived from the well-known ATSP libraries and involve up to 1001 cities, within an hour of CPU time. These are the largest-sized HMATSP instances solved to optimality in the literature.

The fifth problem that we have considered in this dissertation is the fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB), wherein m salesmen distributed among D depots must visit a set of cities and return to their respective depots, while maintaining the number of cities visited by each salesman to be within a prescribed range. We also address an extension of the FD-MTSPB that has not been addressed in the literature, namely the high-multiplicity FD-MTSPB, in which a city is required to be visited multiple times. Novel two-index polynomial-length flow-based formulations, which

are amenable to solution using Benders decomposition, and an exact two-phase approach that relies on solving relaxations of the problem, are developed to solve to optimality the FD-MTSPB and its extension. Computational results reveal that our approaches outperform the exact algorithms available in the literature to solve the FD-MTSPB.

Finally, we analyse the vehicle routing problem with temporary storage (VRP-TS), an extension of the traditional vehicle routing problem, where customers can be utilized as temporary storage locations, i,e vehicles can drop-off extra units that are then picked-up by different vehicle(s). We also study an extension of the VRP, where both split delivery (SD) and temporary storage features are allowed, and denote it as the VRP-SDTS. We present mixed integer programming formulations for the VRP-TS and VRP-SDTS, and study their benefits in reducing the routing cost in comparison with the VRP and VRP with split delivery (VRP-SD). We show that the potential savings of VRP-TS over VRP is at least the same as obtained by the VRP-SD over VRP.

Dedication

To my wife, Josseline, and our little angel, Sebastian.

Acknowledgements

First of all, I would like to express gratitude to my adivor, Dr. Subhash C. Sarin, for all his guidance, patience, and support throughout my Ph.D studies. He is very knowledgeable, dedicated, and caring professor. I started working with Dr. Sarin as an inexperienced graduate student, but he has transformed me into a sharp and independent researcher. I truly thank him for spending tremendous amount of time discussing a wide range of research topics, for reading all my manuscripts and always providing constructive feedback, and for teaching me how to write technically, specially about operations research. Moreover, I thank him for supporting me financially throughout one of his research projects when I needed it the most. Dr. Sarin, you have set an example of excellence as a mentor, researcher, and instructor.

I would like to thank the other committee members for all their help and support during my studies at Virginia Tech. In particular, I would like to thank Dr. John S. Cundiff for teaching me about biomass logistics concepts, and for all his insights in this research. It has been a privilege to work with an upstanding professor like him. He does not just care about research, he also cares about the person. That is invaluable. Also, I want to thank Dr. Barbara M.P. Fraticelli for teaching me optimization theory in different operation research courses she instructed me. I would also like to thank Dr. Douglas R. Bish for working with me at the beginning of my career, and by supporting me throughout this research.

Additionally, I wish to express my gratitude to the Chilean Government, specifically CONICYT, and the Fulbright Comission for awarding me a scholarship to pursue my Ph.D. Their support started with ESL (English as a second language) courses in Chile, and then, in the United States.

I also want to say thank you to my family and friends for their continuous encouragement and support, especially to my father, Froilan Aguayo, and my mother, Irma Bustos, for always being there for me.

Finally, I have to specially thank my wife and best friend, Josseline S. Lozano, for all sacrifices she has made, and by postponing her professional career during these years. Thank "morena" for all your understanding, support, and most importantly, your love. This dream would not be possible without you.

"The LORD is my strength and my shield; my heart trusts in him, and he helps me. My heart leaps for joy, and with my song I praise him." Psalm 28:7.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Biomass logistics

Cellulosic ethanol, produced from energy crops (e.g switchgrass or mischantus) or agricultural residues (e.g, straw or corn stover), is a renewable fuel that mitigates the greenhouse gas (GHG) emissions, and at the same time, reduces the dependency on fossil fuels. Several studies have shown that using cellulosic ethanol instead of gasoline can reduce life cycle GHG emissions by 86%, and recently, nearly 200 nations in the 2015 United Nations Climate Change Conference have signed a historic agreement to cut greenhouse gas emissions. In addition, the Energy Independence and Security Act (EISA) of 2007 in the U.S has established a minimum volume of renewable fuels required for blending into transportation fuel to increase from 9 billion gallons in 2008 to 36 billions gallons by 2022. By that year, 44.4% of the 36 billion gallons of renewable fuel must be produced from cellulosic ethanol. Therefore, it is of national importance to enhance production of cellulosic ethanol, and to make it a viable alternative to fossil fuels.

The availability of biomass for conversion into energy has been demonstrated by several studies. However, a key challenge in generating this energy is to achieve a reduction in feedstock transportation cost, which has been shown to be 35-60% of the total cost of cellulosic ethanol paid at retail pumps [Fales et al. (2008)]. It is, therefore, essential to reduce this cost as much as possible in order to make cellulosic ethanol economically competitive to oil.

The major stages of a supply chain encountered during production of cellulosic ethanol are shown in Figure 1.1. They include production fields, satellite storage facilities (SSLs) or storage locations, bio-energy plants, blending facilities, and retail pumps. The biomass is cultivated and harvested in production fields, which takes place over several months of a year, and the amount of biomass delivered to the SSLs from the farms varies over time. At SSLs, the biomass is hauled to bio-energy plants, where it is processed into ethanol. Afterwards, ethanol is shipped to blending stations, where it is blended with gasoline, and then, it is delivered to the retail pumps. All vehicles with a spark ignition engine can use ethanol blends such as E-10 which is composed of 10% ethanol and 90% gasoline.

Several inherent, well-known optimization problems are encountered during the design and operation of a cellulosic ethanol supply chain, including lot-sizing, location-allocation, and routing problems, among others. At the production field level, the optimization problems faced belong to harvest scheduling, bales collection, and lot-sizing. In the harvest scheduling problem, custom equipment sets must be assigned to harvest production fields that become available dynamically over the planning horizon. This problem can be modeled as an extension of the multiple traveling salesman problem. The bales collection problem, which arises once the fields have been harvested and bales are distributed across the field, consists of designing a set of routes for the in-field equipment for collecting the bales at minimum cost. This problem can be represented as a capacitated traveling salesman problem or vehicle routing problem with unit customer demands. Finally, lot-sizing problems arise because of the need to determine the amount of biomass to produce, to be kept at inventory, and to be delivered during each period.

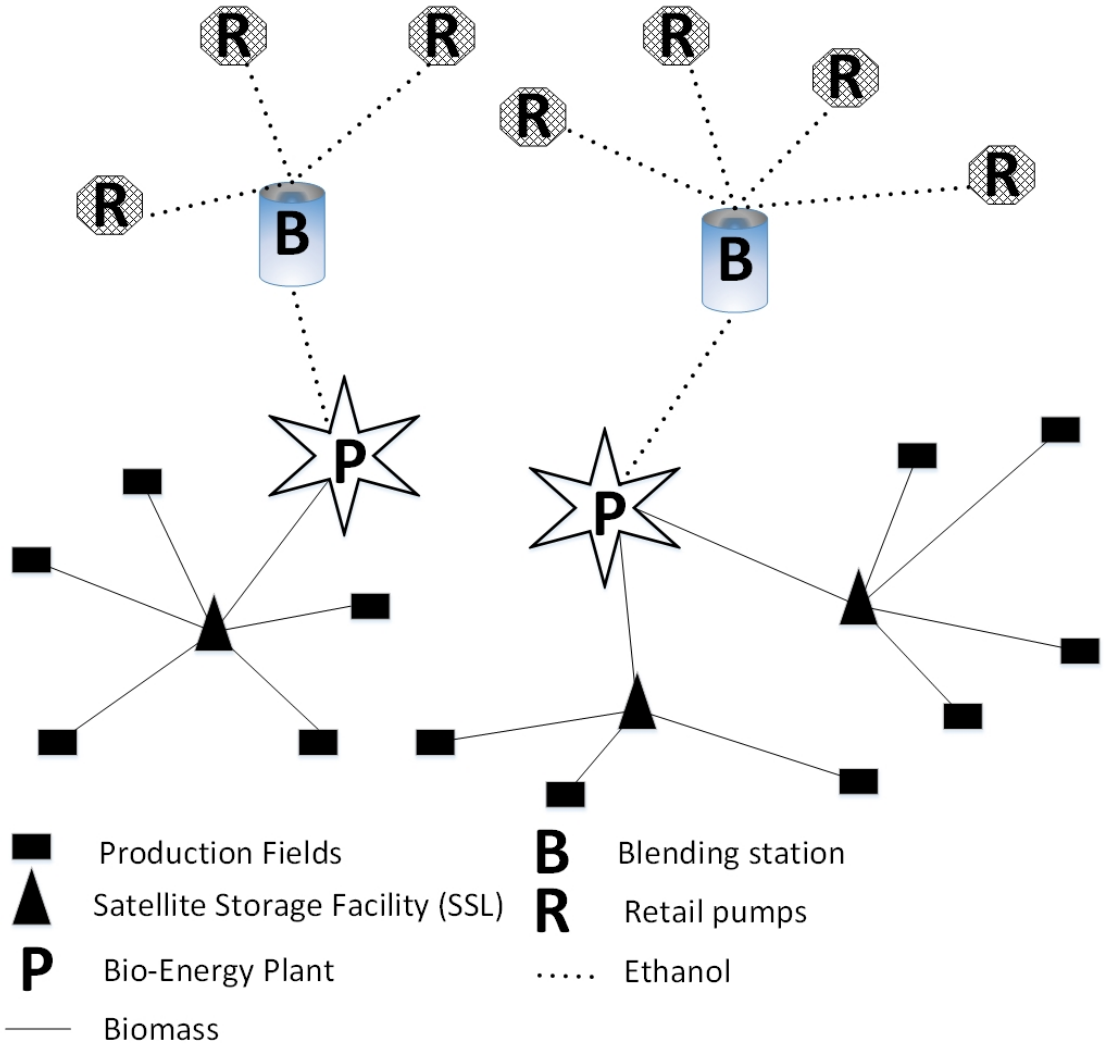

Figure 1.1: Cellulosic ethanol supply chain.

At storage locations, the inherent optimization problems for design of the supply chain pertain to location-allocation, lot-sizing, and routing, among others. The location-allocation decisions correspond to the determination of the number and locations of the SSLs, and the allocation of production fields to these SSLs. Lot-sizing decisions might involve the determination of how much biomass to keep as inventory and to deliver to the bio-energy plants. Routing problems might include the design of routes for mobile equipment to load-out storage facilities or to perform biomass size reduction.

At bio-energy plants and blending facilities, the main optimization problems encountered are the location-allocation, lot-sizing, and routing problems. The location decision corresponds to the location, number, and size of both bio-energy plants and blending facilities. Allocation features corresponds to the assignment of bio-energy plants to blending facilities, and blending facilities to retail pumps. Lot-sizing decisions include how much ethanol to produce, keep in inventory, and deliver in each period to retail pumps. The delivery of ethanol from blending stations to retail pumps can be captured as vehicle routing problems.

In this dissertation, we design a cost-effective biomass logistics supply chain encountered during the production of switchgrass-based ethanol in the Southeast United States, and study a dynamic corn stover harvesting scheduling problem, which is motivated by a real-life case study in the Midwest United States.

Routing problems

Routing problems are combinatorial optimization problems concerned with the design of a set of routes for a fleet of salesmen, or vehicles, in order to satisfy the received request from customers. The most famous routing problems are the traveling salesman, and the vehicle routing problems.

Formally, routing problems can be defined on a graph $G = (V, A)$, where V and A are, respectively, the set of vertices and the arcs connecting the vertices. Next, we present some relevant definitions that will be used in the dissertation.

Definition 1. *A Graph is complete if every pair of distinct vertices is connected. Furthermore, a Graph is undirected and directed, respectively, if every pair of distinct vertices is connected by a unique pair of reverse arcs, and connected by a unique arc.*

Definition 2. *A Path is a route though the graph from vertex to vertex.*

Definition 3. *A Graph is connected if every pair of vertices is connected by some path.*

Definition 4. *The arc distances are symmetric if $c_{ij} = c_{ji}$, and asymmetric if $c_{ij} \neq c_{ji}$,*

$\forall (i,j) \in A$.

Definition 5. *The arc distances satisfy the triangle inequality if* $c_{ik} \leq c_{ij} + c_{jk},\ \forall (i,j)$, (j,k), *and* $(i,k) \in A$.

In this dissertation, we address the following routing problems: the asymmetric traveling salesman (ATSP), the multiple ATSP (mATSP), the high multiplicity ATSP and its extensions, the fixed destination multiple ATSP with load-balancing, and lastly, a generalization of the vehicle routing problem. In the next section, we present formal definitions of these problems.

1.2 Problems addressed in this dissertation and their overreaching relationship

In this section, we present a concise definition of each of the problems studied, and then, present their overreaching relationship. These problems can be classified into two main categories: biomass logistics and vehicle routing.

1.2.1 The biomass logistics supply chain design problem (BL-SCDP)

The major stages of a switchgrass-based bio-ethanol supply chain include production fields, satellite storage facilities (SSLs), and a bio-energy plant. Switchgrass is grown, harvested, and compacted into bales, which are collected from certain production fields. From there, the bales are transported and placed in intermediate facilities for storage or size-reduction. Then, the bales are hauled to the bio-energy plant, where it is transformed into cellulosic ethanol. We designate this problem as the *Biomass Logistics Supply Chain Design Problem* (BL-SCDP).

There are several processes and entities involved in this switchgrass-based bio-ethanol supply chain, shown in Figure 1.2. Entities include production field, switchgrass, balers, round bales, in-field-equipment, racks, load-out equipment, and tractor-trailers. Switchgrass is cultivated for several months, and then is harvested by balers, which cut and compress switchgrass into round bales. The harvesting takes place over several months of a year, and the amount of biomass delivered to the SSLs from the farms might vary over time. Round bales are transported to storage facilities using an in-field-equipment owned by farmers. To unload a storage facility, a load-out equipment set is needed to place bales into racks, which are then hauled by tractor-trailers to the bio-energy plant. Load-out equipment sets are not dedicated to specific storage sites, and they are moved between storages facilities as needed to perform load-out operations.

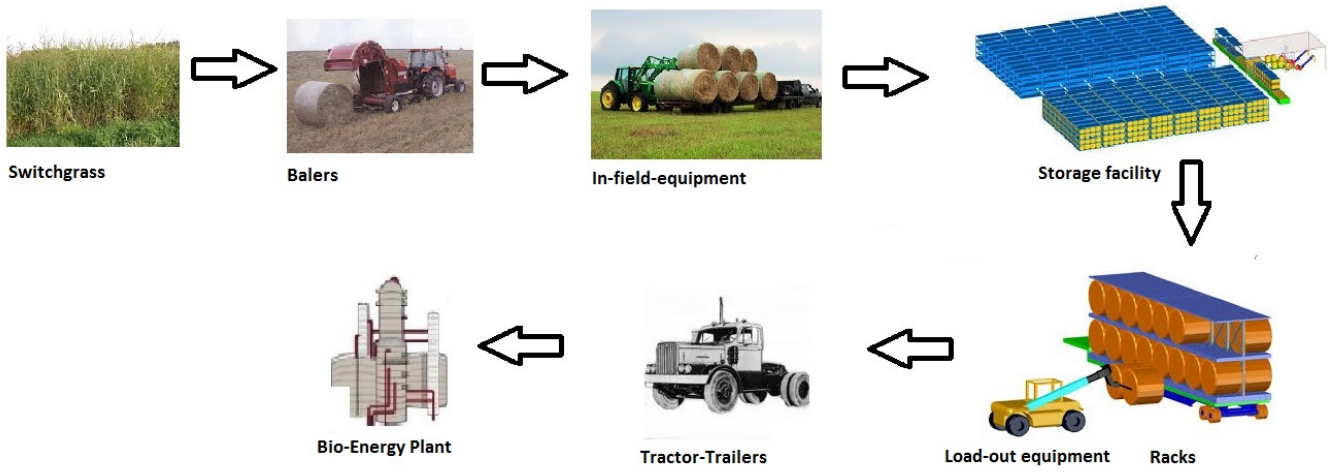

Figure 1.2: Details of a switchgrass supply chain

The BL-SCDP can be defined as follows: *Given a set F of pre-located SSLs, incoming biomass during each time period t at each SSL i, A_i^t, and a planning horizon of length T, determine optimal number of load-out equipment sets (telehandlers), number of tractor-trailers, storage capacity at each SSL, shipping of biomass from SSLs to the bio-energy plant, inventory levels of biomass at the SSLs, routes of load-out equipment sets, and routes of vehicles (tractor-trailers) during each time period, so as to minimize the total cost incurred due to: (a) the*

ownership cost of load-out equipment sets and tractor-trailers, (b) mobilization of the load-out equipment, and (c) hauling cost of biomass from the SSLs to the bio-energy plant.

1.2.2 The corn stover harvesting scheduling problem (CSHSP)

The need for a new source of renewable energy to supply and complement the fossil fuels used for the transportation sector has led to the development of bio-ethanol from agricultural residues such as corn stover, a by-product of corn grain production. Typically, once the grain has been collected, corn stover must be harvested by a third-party company hired by the bio-fuel plant who owns custom, and expensive, harvest equipment sets. This third-party system eliminates the need of corn producers to own and maintain expensive corn stover harvesting equipment. However, the biofuel plant must coordinate and assign these harvest crews to fields that become available dynamically over time. We designate this problem as the corn stover harvesting scheduling problem (CSHSP).

Each custom harvester has a set of equipment needed to harvest the field and place the stover in a roadside storage. For the sake of convenience, we refer to the set of equipment needed to harvest a field as a "baler", and also refer to the cellulosic plant as a "depot". Figure 1.3 presents pertinent features of the dynamic CSHSP, where a unique baler is located at the central depot denoted by 1 (square), and three fields (2, 3, and 4, denoted by circles) are ready to be harvested at time 0. The required number of harvest days for Fields 2, 3, and 4 are $n_2 = 1$, $n_3 = 2$, and $n_4 = 1$, respectively. They are calculated for each field as follows: field area (hectares) $\times$ expected corn stover yield (Mg/hectare) divided by the baler capacity (Mg/day). Furthermore, fields have time windows, $w_f = [r_f, d_f]$, that specify the periods when they can be harvested, where r_f is the ready time (call-in time) for field f and d_f is the due date. In this example, we have $w_2 = [1, 5]$, $w_3 = [1, 7]$, and $w_4 = [1, 6]$. The goal is to determine a route for the baler that minimizes the total routing cost and the due-date-related-penalties for late harvest (the contract with the farmer assigns a penalty if

the stover is not harvested within a given number of days after the call-in date). The optimal schedule for day 1 (period 1), shown in Figure 1.3 (a), is: 1(0)-2(1)-3(2)-3(3)-4(4)-1(5), where notation $f(t)$ denotes that the baler is located at field f in day t. Now, at the end of day 1, two new fields, denoted by 5 and 6, arrive with their required number of harvest days given by $n_5 = 1$ and $n_6 = 1$, and their time windows given by $w_5 = [2, 8]$ and $w_6 = [2, 8]$. Given these new arrivals, and considering that Field 2 is already harvested, and the baler is located at this field, the problem is reoptimized so that Fields 5 and 6 are incorporated in the proposed route. The new route generated is: 1(0)-2(1)-5(2)-6(3)-3(4)-3(5)-4(6)-1(7) and it is shown in Figure 1.3 (b). Note that the baler first visits the new arrivals (Fields 5 and 6), and then, it continues with its original route (Fields 3 and 4). Capacities of the balers are different and this is considered in the analysis. Furthermore, multiple balers of each technology are available, and a given field can be visited by multiple balers of the same technology (round or rectangular) at the same time.

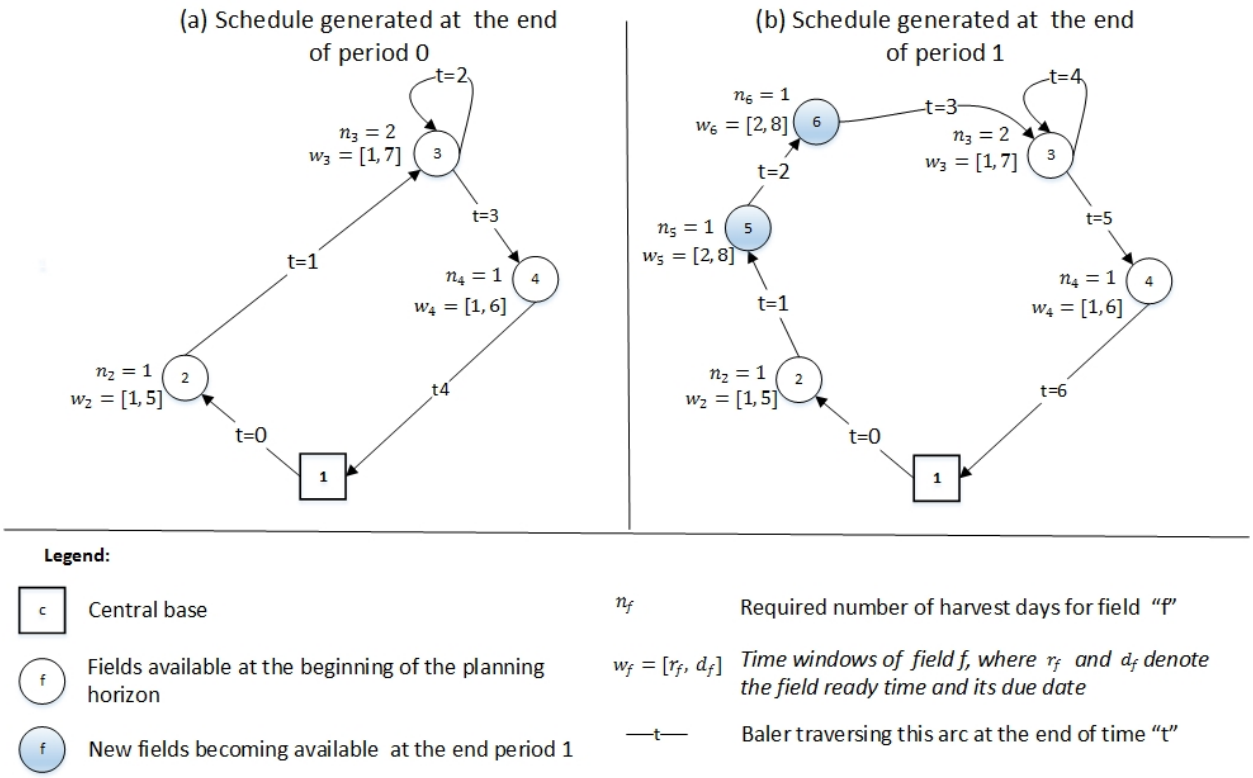

Figure 1.3: Features of the CSSHP.

The static corn stover harvesting problem (CSHSP) can be concisely defined as follows: Given a set of F pre-located productions fields, a_f Mg of corn stover available in field $f \in F$, a time window, $w_f = [r_f, d_f]$, where r_f and d_f are the ready time and due date of field f, B types of balers with capacity k_b (Mg per period), and a planning horizon of length T, determine the optimal number of balers, the type of bales to produce in field $f \in F$, when and the required number of periods to complete the harvest of field f, and, the routing of balers over the planning horizon. The goal is to minimize the total cost incurred by the plant. The contract paid to the harvesting contractor pays them a fixed cost per Mg for all stover they harvest. This total cost is the sum of the following cost: (a) Capital cost due to corn stover harvesting equipment, (b) contract price ($/Mg) paid to custom harvesters by the plant, (c) due-date-related penalties paid to the feedstock producers, and (d) total routing cost for all baling contractors.

The definition of the dynamic CSHSP is identical to the static CSHSP, except that the fields become available dynamically over the planning horizon (fields are called in each day over the harvesting season). It is assumed that the number of balers is known *a priori*, which is determined at the beginning of analysis by solving the static CSHSP by taking into consideration all the fields under contract by the plant.

1.2.3 The single and multiple-asymmetric traveling salesman problems (ATSP and mATSP)

The asymmetric traveling salesman problem (ATSP) is one of the most well-known combinational optimization problems studied in the literature, and hundred of papers and several books have been devoted to its study. It consists of determining a tour that starts and ends at a given base city after having visited a set of cities exactly once. Practical applications of the TSP include the drilling of printed circuit boards, X-ray crystallography, overhauling gas turbine engines, order-picking in warehouses, computer wiring, clustering of a data array,

vehicle routing, and scheduling, among others (Reinelt (2003)).

An extension of the ATSP, which has not received much attention, is the multiple asymmetric traveling salesmen problem (mATSP), where multiple salesmen are located at the depot. Some applications of this problem reported in the literature include print press scheduling, crew scheduling, school bus routing problem, interview scheduling, hot rolling scheduling, and design of global navigation satellite surveying network (Bektaş (2006)).

Figures 1.4 (a) and (b), respectively, present instances of the ATSP and mATSP. In this example, 1 denotes the depot (square) where m salesmen are located, and they must visit a set of cities denoted by $2, 3, 4$, and 5 (circles). In the ATSP solution displayed in Figure 1.4.(a), a unique salesman is located at the depot ($m_1 = 1$) and the optimal tour is 1-2-3-4-5-1. In the mATP solution, shown in Figure 1.4.(b), two salesmen are located at the depot ($m_1 = 2$), and their tours are 1-2-3-1 (denoted by solid line) and 1-4-5-1 (dash line).

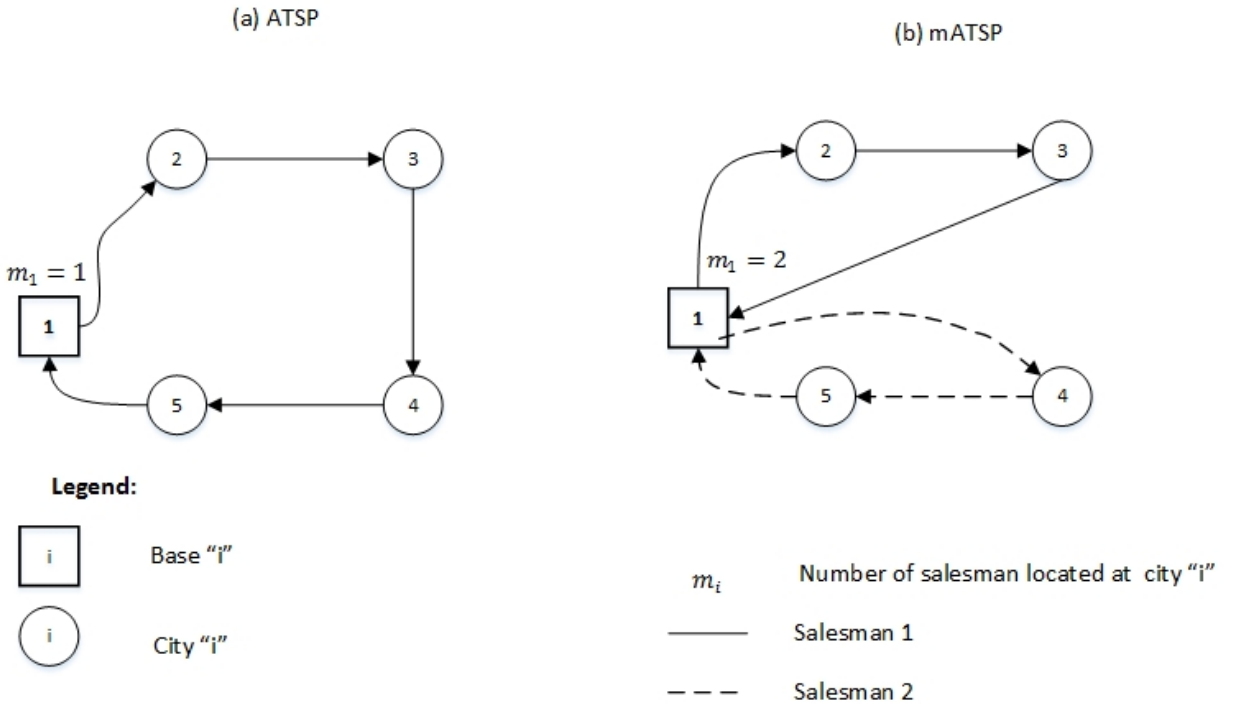

Figure 1.4: Asymmetric traveling salesman problem (ATSP) and multiple ATSP examples.

The ATSP can be defined as follows: *Given a complete graph with vertex set N in which city*

(1) *denotes the base city, and an asymmetric distance matrix* $[c_{ij}]$, $i, j \in N$, $j \neq i$, *determine an optimal tour that starts and ends at a base city after having visited city* i *exactly once,* $\forall i \in N$, *while minimizing the total distance traveled.*

The mATSP can be defined as follows: *Given a complete graph with vertex set* N *in which city* 1 *denotes the base city, an asymmetric distance matrix* $[c_{ij}]$, $i, j \in N$, $i \neq j$, *and* m *salesmen located at the base city, determine* m *tours that start and end at the base city after collectively having visited city* i *exactly once,* $\forall i \in N$, *while minimizing the total distance traveled.*

1.2.4 The high-multiplicity asymmetric traveling salesman problem (HMATSP) and its extensions

The high-multiplicity asymmetric traveling salesman problem (HMATSP) is a generalization of the traveling salesman problem, where a salesman departs and then returns to the base city after having visited each city multiple times. Applications of the HMATSP include the scheduling of landing airplanes (Cosmadakis and Papadimitriou (1984)), the CHES problem (Baker and Muckstadt (1989)), the primary pharmaceutical manufacturing scheduling problem (Sarin et al. (2014b)), and the equipment fleet management in biomass logistics (Chapter 2).

We also study two extensions of the HMATSP that involve multiple salesmen and bases. In the first variant, multiple salesmen start from a single base and visit each city a specified number of times before returning to their starting points. We denote this problem as the *single-base multiple salesmen HMATSP* (SB-HMmATSP). In the second variant, multiple salesmen start from multiple bases and visit each city a specified number of times before returning to starting base city. Note that the salesmen are not forced to return to their starting bases. We denote this problem as *the multiple-base, non-fixed destination, multiple*

salesmen HMATSP (MBNF-HMmATSP).

Figure 1.5 illustrates the HMATSP and its extensions addressed in this dissertation. In this example, the input parameters are as follows: $n = 5$ (cities), the cost matrix $[c_{i,j}]$={ 1, 2, 5, 7, 8; 4, 8, 7, 8, 8 ; 4, 8, 8, 7, 8 ; 1, 4, 0, 1, 8; 1, 7, 8, 6, 1}, and $n_i = \{1, 1, 2, 1, 1\}$. The number of salesmen located at each base is denoted by m_i. Figure 1.5(a) presents an optimal solution for the HMATSP, where node 1 is the base (square), a unique salesman is located at node 1 ($m_1 = 1$), and the optimal routing cost is 25. Figure 1.5(b) depicts an optimal solution for the single-base multiple salesmen HMATSP (SB-HMmATSP), where as before, the base city is node 1, but now two salesmen are located at this base ($m_1 = 2$, with their respective routes designated by solid and dashed lines), and where the total routing cost incurred is 27. Figure 1.5(c) displays a solution for the multiple-base, non-fixed destination, multiple salesmen HMATSP (MBNF-HMmATSP), where nodes 1 and 2 (squares) are the bases from where the salesmen ($m_1 = 1$ and $m_2 = 1$) begin their tours, and the total routing cost incurred is 29. Also, note that the salesmen do not return to their respective bases in the solution to this instance, which is allowed.

The HMATSP can be defined as follows: *Given a complete graph with vertex set N, an asymmetric distance matrix* $[c_{ij}]$*, and positive integers* n_i*,* $i, j \in N$*, determine an optimal tour that starts and ends at a base city after having visited city* i *exactly* n_i *times,* $\forall i \in N$*, while minimizing the total distance traveled.*

The SB-HMmATSP is a natural extension of the HMATSP wherein m salesmen are located at a single base and can visit cities multiple times, and is defined as follows: *Given a complete graph with vertex set N, an asymmetric distance matrix* $[c_{ij}]$*, and positive integers* n_i*,* $\forall i \in N$*, find* m *tours that start and end at the given base after having visited node* i *exactly* n_i *times,* $\forall i \in N$*, such that the total distance traveled is minimized.*

The MBNF-HMATSP is another extension of the HMATSP, which is stated as follows: *Given a complete graph with vertex set* $N = B \cup N'$*, where the first* $|B|$ *nodes of N comprise*

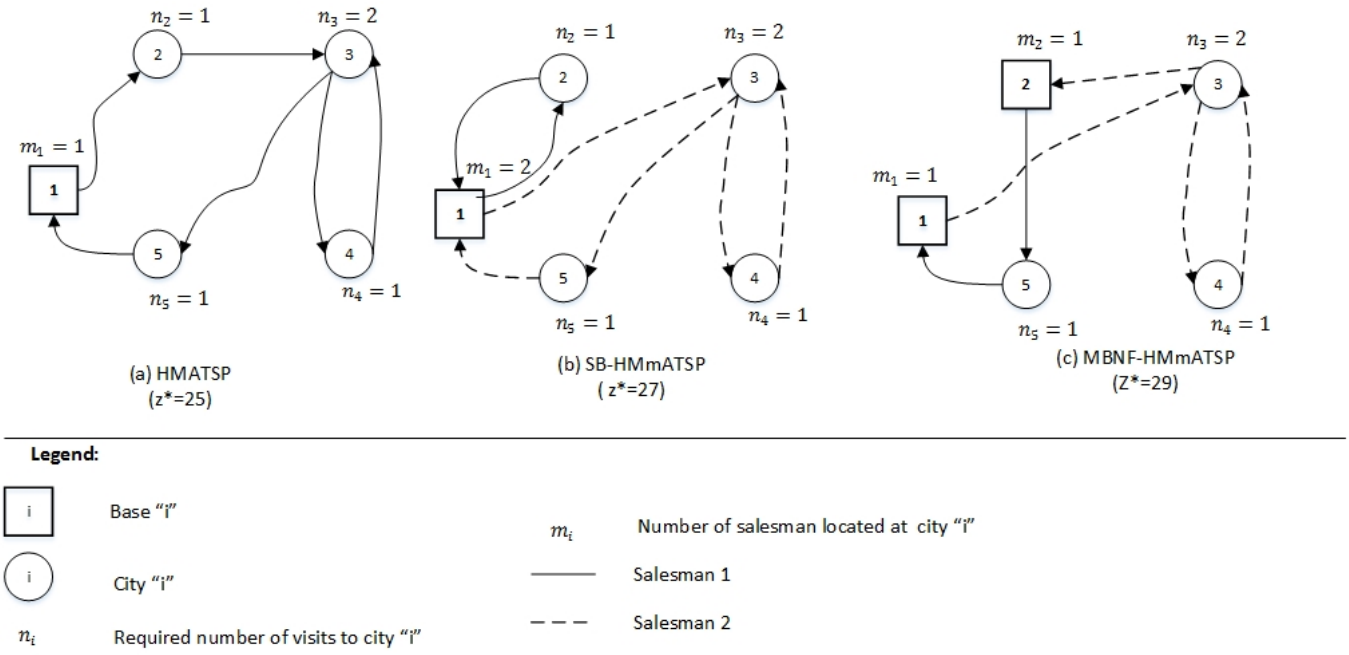

Figure 1.5: The HMATSP and its extensions.

a set of base cities having m_b salesmen located initially at base $b \in B$, and where $N' = \{|B|+1, |B|+2, ..., n\}$ comprise the set of cities to be visited, and given an asymmetric distance matrix $[c_{ij}]$, and positive integers n_i, $\forall i \in N$, find $m = \sum_{b \in B} m_b$ tours, with m_b tours starting at the base city $b \in B$, and likewise, with m_b tours ending at a base city $b \in B$, where any such tour can start and end at different base cities while collectively having visited city i exactly n_i times, $\forall i \in N$, so that the total distance traveled is minimized.

1.2.5 The fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB)

The fixed-destination multi-depot traveling salesmen problem is an extension of the multiple asymmetric traveling salesman problem (mATSP), where the salesmen located at different bases depart, and then, return to their starting points after having visited a set of cities. Example of fixed destination routing include variants of the vehicle routing problems (Golden

et al. (1977) and Laporte et al. (1988)), the multiple traveling salesmen problem (Kara and Bektaş (2006) and Burger (2014)), and postal distribution, less-than-truckload transport operations, the traveling repairmen problem, and balance billing-cycle vehicle routing problems (Bektaş (2012)).

The fixed-destination multi-depot traveling salesmen problem with load balancing (FD-MTSPB) has been introduced by Bektaş (2012), where the number of cities visited by each salesman must lie in a given interval so that there exists a balance workload among salesmen. We also study an extension of the FD-MTSPB, where cities might be visited multiple times and denoted it as high multiplicity FD-MTSPB (HMFD-MTSPB).

Figure 1.6 illustrates the problems addressed in this research, where the number of salesmen located at each base is denoted by m_i, and the number of visits required at each city is given by n_i. Figure 1.5(a) presents an optimal solution for the fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB), where node 1 and 2 are the bases (squares) from where the salesmen ($m_1 = 1$ and $m_2 = 1$) begin and end their tours, and the range of cities visited by each salesman is between [2, 3]. Note that in this problem each city is required to be visited once, i.e, $n_3 = n_4 = n_5 = n_6 = n_7 = 1$. Figure 1.5(b) shows an optimal route for the high-multiplicity MTSPB (HMFD-MTSPB), where cities are required to be visited multiple times. Again node 1 and 2 are the bases (squares) from where the salesmen ($m_1 = 1$ and $m_2 = 1$) begin and end their tours, and the range of cities visited by each salesman is between [2, 5]. The number of visits to each city is given by: $n_3 = n_5 = n_6 = n_7 = 1$ and $n_4 = 2$.

The FD-MTSPB can be defined as follows: *Given a complete graph with vertex set* $N = D \cup N'$*, where the first* $|D|$ *nodes of* N *comprise a set of depots having* m_d *salesmen located initially at depot* $d \in D$*; and where* $N' = \{|D|+1, |D|+2, ..., n\}$ *comprises the set of cities to be visited, and given an asymmetric distance matrix* $[c_{ij}]$*, find* $m = \sum_{d \in D} m_d$ *tours, with* m_d *tours starting and ending at depot* $d \in D$*, while collectively having visited city* i *exactly once,*

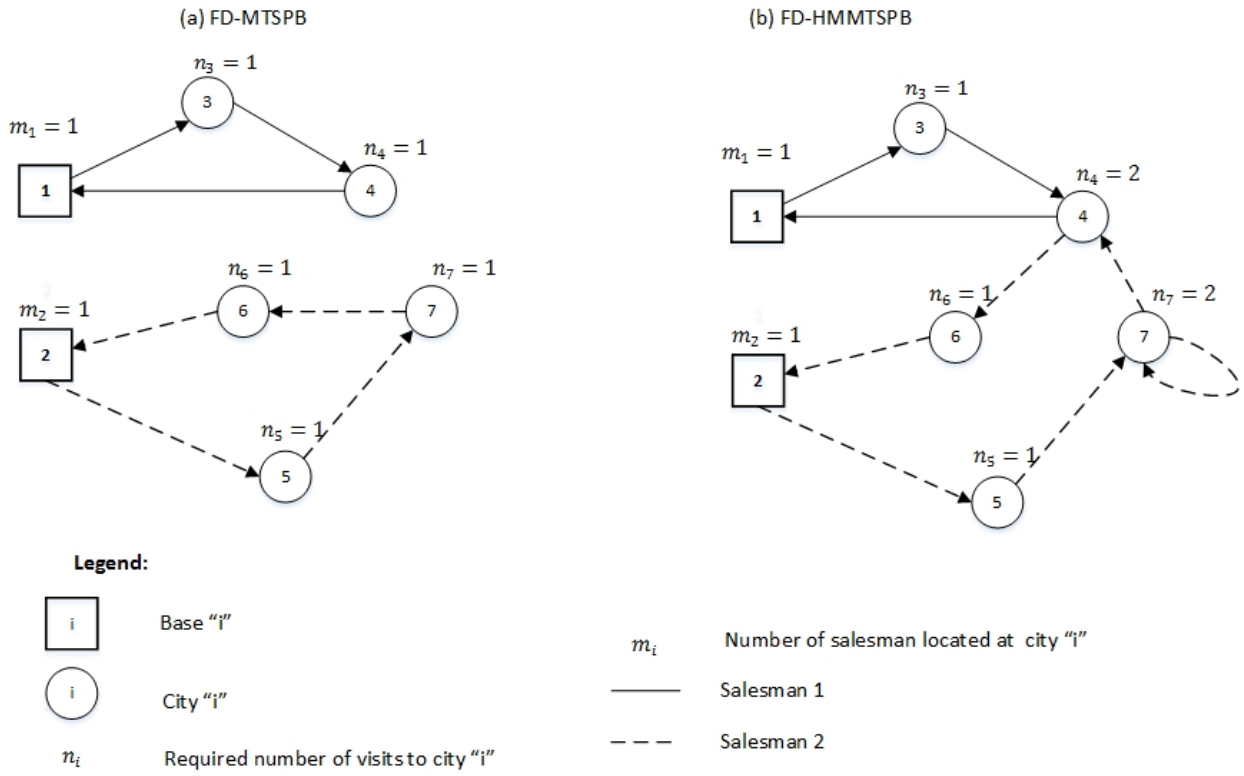

Figure 1.6: The FD-MTSPB and its extensions.

$\forall i \in N'$, *such that the total distance traveled is minimized and the number of visits made by each salesman lies in the interval [L,K], where L and K are specified positive integers.*

The high multiplicity FD-MTSPB (HMFD-MTSPB) can be defined as follows: *Given a complete graph with vertex set* $N = D \cup N'$, *where the first* $|D|$ *nodes of N comprise a set of depots having* m_d *salesmen located initially at depot* $d \in D$, *and where* $N' = \{|D|+1, |D|+2, ..., n\}$ *comprises the set of cities to be visited, and given an asymmetric distance matrix* $[c_{ij}]$, *and a positive integer number* n_i *for each* $i \in N'$, *find* $m = \sum_{d \in D} m_d$ *tours, with* m_d *tours starting and ending at depot* $d \in D$, *while collectively having visited city i exactly* n_i *times,* $\forall i \in N'$, *such that the total distance traveled is minimized and the number of visits made by each salesman is between a given interval denoted by [L,K], where L and K are specified positive integers.*

1.2.6 The vehicle routing problem with temporary storage (VRP-TS)

The vehicle routing problem (VRP) is a classical combinatorial optimization problem in which vehicles make deliveries to a set of customers with known demand and each customer is visited by a unique vehicle. It has been shown that BY allowing a customer to be served by more than one vehicle, an extension known as the the VRP with split delivery (VRP-SD), the routing cost can reduced. Moreover, a tight bound has been proven for this reduction.

Consider an extension of the VRP, where a customer can be used as a temporary storage, that is, vehicles can drop-off more than the customer's demand at a location to be picked by another vehicle (vehicles) later for delivery to another customer. We denote this problem as the VRP with temporary storage (VRP-TS). Figure 1.7 presents difference between the traditional VRP and VRP-TS. Let z(VRP) and z(VRP-TS) denote the optimal cost of VRP and VRP-SD solutions, respectively. A fleet of homogeneous vehicles with capacity 5 is located at the depot, denoted by a square, and customers, represented by circles, have demands indicated by the numbers inside these circles. In the VRP solution, shown in Figure 1.7.b., three vehicles are utilized, each customer is visited with a back and forth trip from the depot, and the cost incurred is z(VRP) = $6M$. Meanwhile, in the VRP-TS solution, Figure 1.8.c, two vehicles are used, where one vehicle, denoted by a dash line, visits the customer in the middle with a back and forth trip from the depot, and uses this location as a temporary storage in which 1 unit is left temporarily. The other vehicle, denoted by solid line, starts in the left corner of the figure, and then, it drives to pick up 1 unit left at the temporary storage. Finally, this vehicle satisfies the demand of the unvisited customer. Thus, z(VRP-TS) = $4M + 2/M$. By letting M going to infinity, the ratio z(VRP)/z(VRP-TS) goes to 3/2. Thus, temporary storage can be advantageous in reducing the number of vehicles and the routing cost when compared with the VRP.

Figure 1.8 presents difference between the VRP-SD and VRP-TS. In this example, we fix

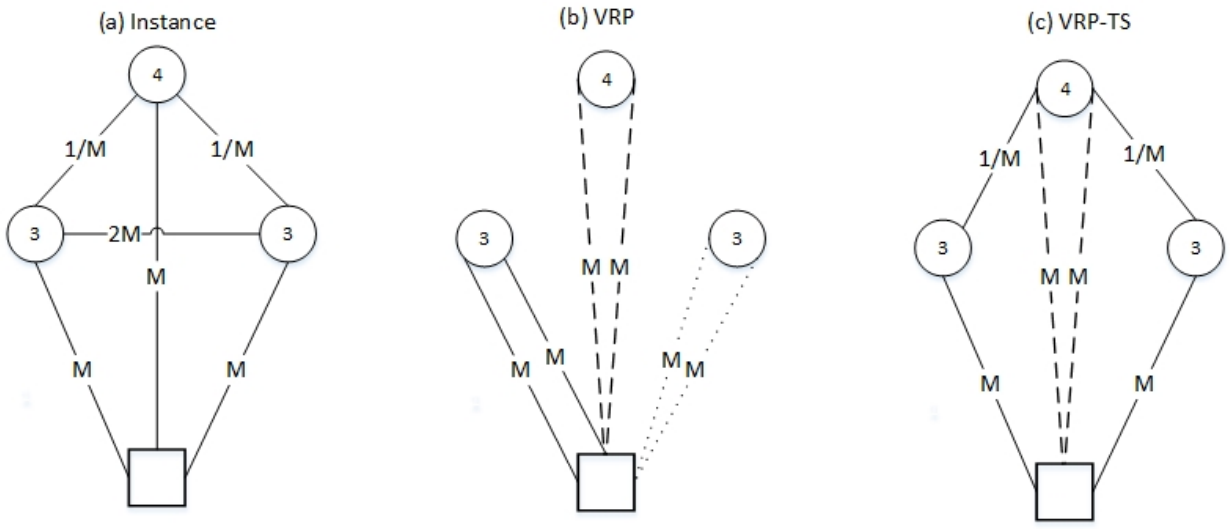

Figure 1.7: Differences between VRP and VRP-TS.

the number of vehicles to the minimum, which is calculated by $\lceil \sum_{i \in N'} d_i / C \rceil$, where N' denotes the set of customers to be visited with their demands given by d_i, $i \in N - \{1\}$ and C represents the vehicle capacity. Thus, the minimum number of vehicles, in this example, is $\lceil \sum_{i \in N'} d_i / C \rceil = (2 + 5 + 5 + 4)/8 = 2$. Let z(VRP-SD) denote the optimal cost of a VRP-SD solution. The VRP-SD solution, Figure 1.8.b, splits the demand of a customer, and its routing cost is z(VRP-SD) $= 8M + 1/M$. On the other hand, in the VRP-TS solution, Figure 1.8.c, one vehicle (denoted by a solid line) picks 1 unit left temporarily by the other vehicle (denoted by dashed line) at the closest customer location from the depot, and its routing cost is z(VRP-TS) $= 6M + 1/M$. By letting M go to infinity, the ratio z(VRP-SD)/z(VRP-TS) goes to 3/2. Therefore, temporary storage is different from split delivery, and it can be advantageous in reducing the routing cost as well.

The vehicle routing with temporary storage (VRP-TS) can be defined as follows: *Given a complete graph with vertex set* $N = 1 \cup N'$, *where* 1 *denotes the depot in which a fleet of homogeneous vehicles with capacity* C *is located,* $N' = \{2, ..., n\}$ *comprises the set of customers with a demand of* d_i, $d_i \leq C$, $\forall i \in N'$, *and given a symmetric distance matrix* $[c_{ij}]$, *find a set of routes that satisfy the demand of all the customers while minimizing*

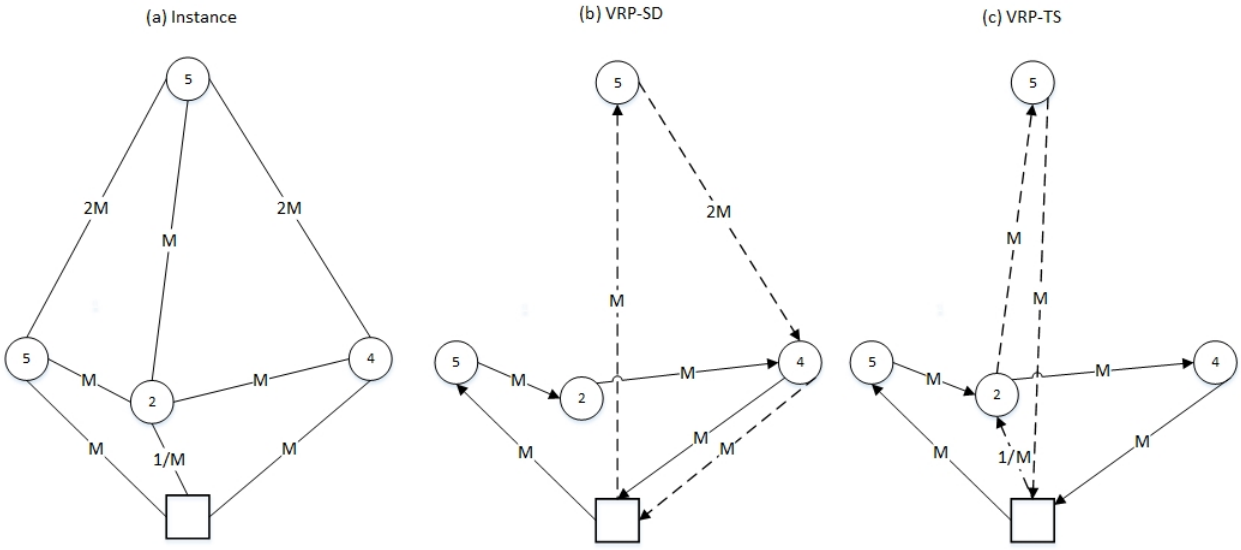

Figure 1.8: Differences between VRP-SD and VRP-TS.

the total distance traveled by all the vehicles, assigning each vehicle to a unique route, and permitting customers locations to be used as temporary storages.

We are also interested in combining both split delivery and temporary storage, and study their synergy. We denote this problem as VRP-SDTS.

1.2.7 Overreaching relationship among the problems addressed in this research

The relationship among the problems addressed in this dissertation is established in Figure 1.9, where a dotted arrow from problem 1 (p_1) to problem 2 (p_2) denotes that either p_2 is a special case of p_1 or p_2 arises as a subproblem in p_1. First, in the biomass logistics area, we study a biomass logistic supply chain design problem (BL-SCDP) (Chapter 2), and a dynamic corn stover harvesting scheduling problem (CSHSP) (Chapter 3). In the routing area, we addresses the asymmetric traveling salesmen problem (ATSP) and multiple ATSP (mATSP) (Chapter 4), the high multiplicity ATSP (HMATSP) (Chapter 5), the fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB)

(Chapter 6), and the vehicle routing problem with temporary storage (VRP-TS) (Chapter 7). The relationship between Chapters 2 and 5 can be established by noticing that in the BL-SCDP, the routing of load-out equipment can be modeled as an extension of the HMATSP. Similarly, the relationship between Chapters 2 and 7 is established by realizing that in the BL-SCDP, the daily schedule of tractors-trailers to pick up bales from storage can be captured as a vehicle routing problem with temporary storage.

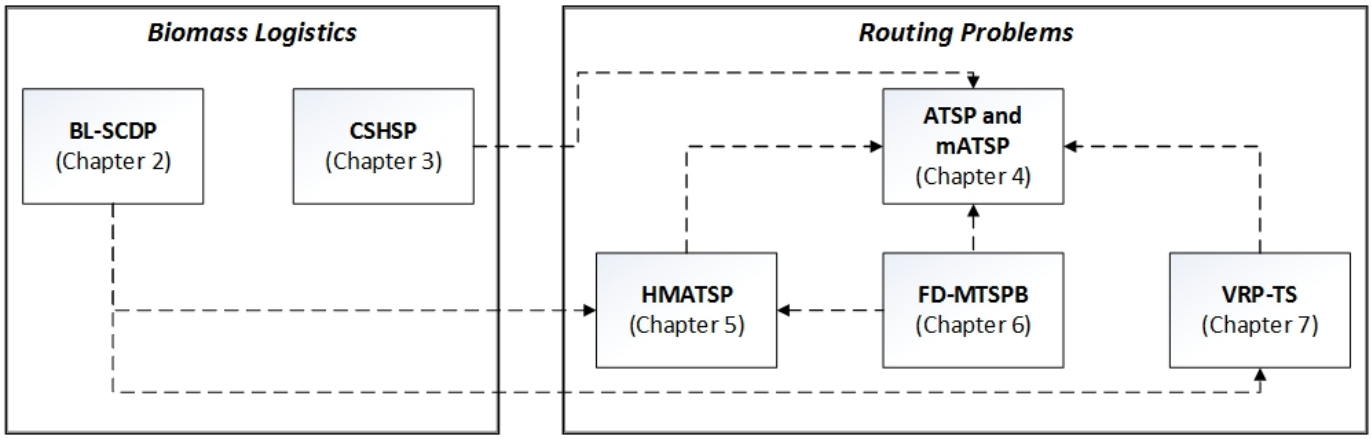

Figure 1.9: Relationship among the problems addressed in this dissertation.

The relationship between Chapters 3 and 4 is established by noticing that the routing of harvesting equipment sets is an extension of the traveling salesman problem. In addition, the HMATSP (Chapter 5), the FD-MTSPB (Chapter 6), and the VRP-TS (Chapter 7) are all extensions of the ATSP and multiple mATSP (Chapter 4). Finally, FD-MTSPB (Chapter 6) is connected to HMATSP (Chapter 5) since the former chapter also investigates the high-multiplicity FD-MTSPB.

1.3 Research objectives and contributions

In this section, we present the objectives and contributions of this research, which are classified into two main areas: biomass logistics and routing problems.

Biomass logistics area

The aim of this dissertation is to model, analyse, and develop exact algorithms to solve some biomass logistics supply chain problems encountered in practice in order to obtain cost-effective solutions. Specific objectives are to:

1. Develop an effective exact algorithm to solve a biomass logistics supply chain design problem encountered in practice.
2. Model the operation of load-out equipment sets for unloading SSLs as an extension of the high-multiplicity traveling salesman problem.
3. Model and develop novel solution approaches for the corn stover harvest scheduling problem in a dynamic environment.

Contribution of this research

1. We introduce a new and challenging BL-SDCP that lies at the interface of three areas: single-item capacitated lot-sizing problem with multiple parallel facilities (CLSP-MF), time-dependent selective high-multiplicity multiple travelling salesmen problem (TDS-HMmATSP), and biomass logistics (BL).
2. We develop an effective branch-and-price algorithm to solve a real-world biomass supply chain design problem.
3. We apply our methodology to a real-life switchgrass-based-ethanol logistics supply chain.
4. We introduce a second problem, the corn stover harvest scheduling problem (CSHSP), to the literature that lies at the interfaces of the time-dependent, high-multiplicity, and multiple heterogeneous traveling salesmen problems in the presence of a time window in which to harvest each field.

5. We propose a mixed-integer programming formulation to model the CSHSP, and develop an effective approach to solve the dynamic CSHSP that can be used in practice.

6. We develop several insights for use by the "Feedstock Manager" at a bio-fuel plant to effectively coordinate harvesting of corn stover.

Routing area

Research objectives

The main objective is to develop new compact formulations and effective exact algorithms for different routing problems with emphasis on problems that arise in the biomass logistics area. Specific objectives are to:

1. Develop effective solution approaches for the ATSP and mATSP.

2. Propose new polynomial-length formulations and effective exact algorithms for the HMATSP and its extensions.

3. Study new polynomial-length formulations, and develop effective solution approaches for the FD-MTSPB and its extensions.

4. Propose new strategies to reduce the cost incurred in routing of vehicles.

Contribution of this research

1. We propose an effective implementation of the idea of generating subtour elimination constraints (SECs) from integer solutions for the solution of ATSP and mATSP, which is able to solve real-world ATSP instances and competes favorably with the most effective state-of-the-art exact algorithms developed for the ATSP in the literature. Its additional advantage is that it can be implemented using an off-the-shelf software like CPLEX, and it does not rely on any specialized algorithm.

2. We solve large-sized mATSP instances to optimality for the first time in the literature.

3. We present a new formulation for the HMATSP, which is faster than the direct solution, by a commercial solver (CPLEX), of the existing compact formulations.

4. We study several extensions of the HMATSP that have not been addressed in the literature, and we propose model formulations for these problems.

5. We develop exact algorithms for the solution of the HMATSP, and its extensions, that can solve to optimality (within an hour of CPU time) instances having up to 1001 cities that are taken from a well-known TSP library, and that outperform the direct solution of these model formulations obtained using CPLEX (version 12.5.1)

6. We present a new polynomial-length formulation for the fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB).

7. We propose a state-of-the-art exact algorithm to solve the FD-MTSPB, which outperforms the best algorithms reported for this problem in the literature.

8. We introduce new model formulations for the high multiplicity FD-MTSPB, which has not been addressed in the literature.

9. We introduce the vehicle routing problem with temporary storage (VRP-TS), and the vehicle routing problem with split delivery and temporary storage (VRP-SDTS).

10. We propose polynomial-length formulations and develop structural properties of both VRP-TS and VRP-SDTS.

1.4 Organization of this work

The remainder of this dissertation is organized as follows. Chapter 2 introduces the BL-SCDP, an integrated model that lies at the interface of the single-item capacitated lot-

sizing problem with multiple parallel facilities and time-dependent selective high-multiplicity multiple traveling salesmen problem. Chapter 3 addresses a corn stover harvest scheduling problem, where fields ready to be harvested become available dynamically over the planning horizon and a set of harvesting equipment must be scheduled to visit and collect biomass from these fields. Chapter 4 is devoted to study the well-known asymmetric traveling salesman (ATSP) and multiple asymmetric traveling salesman (mATSP), where a new and effective effective approach is proposed to solve large-sized ATSP and mATPS instances to optimality. Chapter 5 introduces a new polynomial-length formulation and exact algorithms for the high-multiplicity asymmetric traveling salesman problem (HMATSP) and its extensions. Chapter 6 proposes a new formulation and exact approaches for the fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB), wherein m salesmen distributed among D depots must visit a set of cities and return to their respective depots, while maintaining the number of cities visited by each salesman to be between a prescribed range. It also addresses the high-multiplicity FD-MTSPB, which has not been studied in the literature. Chapter 7 introduces the vehicle routing problem with temporary storage (VRP-TS), an extension of the vehicle routing problem, where customers are utilized as temporary storage points. Finally, concluding remarks and suggested directions for future work are presented in Chapter 8.

Chapter 2

A Branch-and-Price Approach for Biomass Logistic Supply Chain Design Problem (BL-SCDP)

2.1 Introduction

The major stages of a switchgrass-based bio-ethanol supply chain encountered during production are shown in Figure 2.1. They include production fields (squares), satellite storage facilities (SSLs) (triangles), and a bio-energy plant (star). The switchgrass is cultivated and harvested in production fields. The round bales are collected from the fields and transported by farmers using in-field transportation to the SSLs for storage. The harvesting takes place over several months of a year, and the amount of biomass delivered to the SSLs from the farms varies over time. We assume the duration over which the biomass is delivered to the SSLs from the farms to be split into T time periods and the amount of biomass delivered to each SSL during each time period to be known *a priori*. At SSLs, the round bales are picked up by mobile equipment and placed in 20-bale racks that are then hauled by tractor-trailers

to a bio-energy plant. In this chapter, we develop a branch-and-price-based methodology to effectively determine optimal number of load-out equipment sets (for loading-out an SSL) and tractor-trailers (for hauling bales to the bio-energy plant), capacities of SSLs, inventory levels of biomass at the SSLs, and routings for both load-out equipment sets and tractor-trailers during each time period. We designate this problem as the *Biomass Logistics Supply Chain Design Problem* (BL-SCDP).

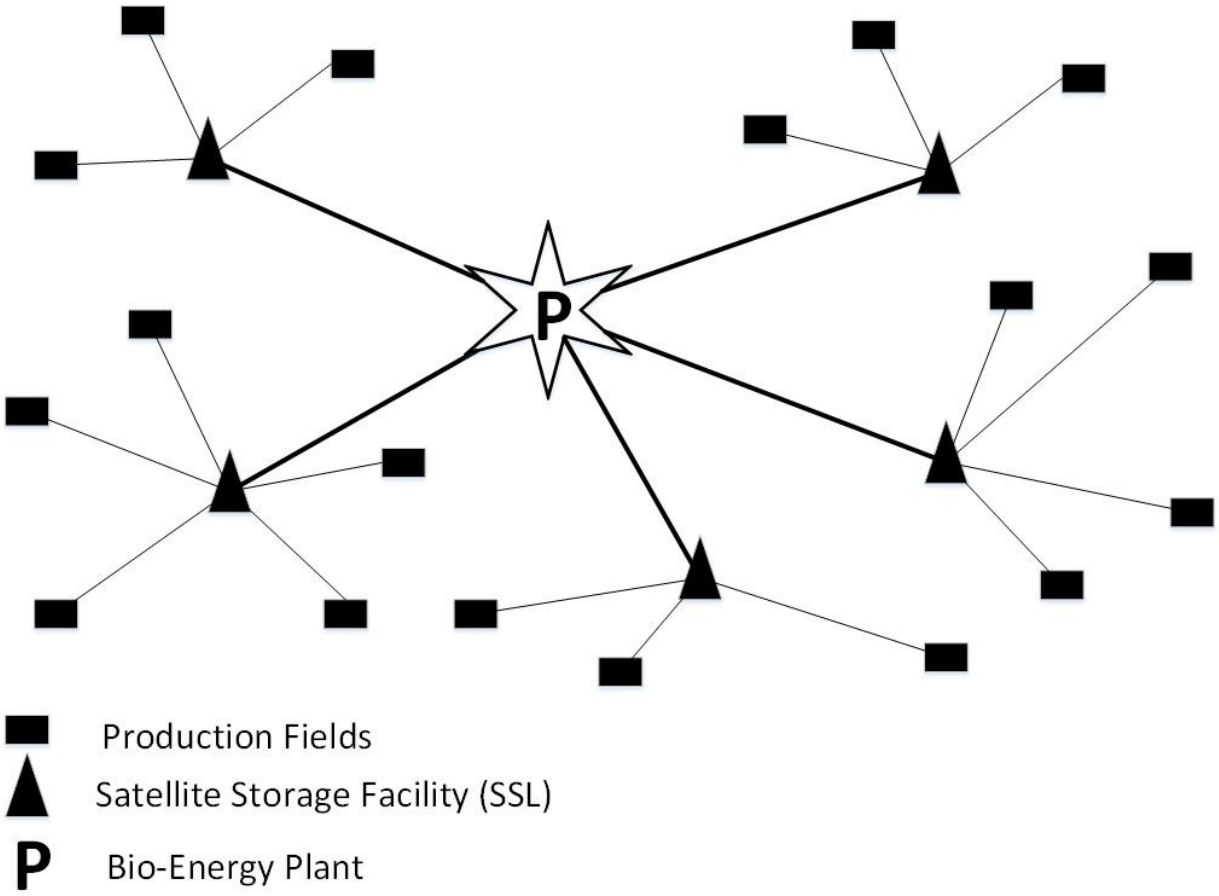

Figure 2.1: Switchgrass supply chain.

We have organized the chapter as follows: In Section 2.2, we present a brief description of the biomass logistics supply chain under study, and also, present a brief review of literature related to our problem. In Section 2.3, we formulate the BL-SCDP as a mixed integer programming model and derive some valid inequalities. Section 2.4 contains the proposed branch-and-price approach along with the details of its implementation to solve the BL-

SCDP. Computational results of applying this methodology to problem instances inspired by a real-life switchgrass-based logistics supply chain are presented in Section 2.5.

2.2 System description, problem statement, and literature review

We assume the switchgrass to be produced and managed by feedstock producers. After baling, the feedstock producers transport the biomass to SSLs. Each SSL is an uncovered site that is accessible by tractor-trailers, and it is close to main highways or secondary roads for easy transportation of biomass by trucks to the bio-energy plant. We focus on operations to haul the 1.5 m diameter, 0.4 Mg (15% moisture content w.b.) round bales to the bio-energy plant. Round bales are commonly used in the Southeastern United States because of their ability to shed water in single-layer ambient storage. Next, we briefly describe the operations involved in the biomass logistics system that we consider.

As alluded to earlier, all production-field activities, such as cultivation, harvest, and transportation of biomass from cultivation sites to storage locations are performed by feedstock producers. The costs associated with all field-related activates are included in a contract price paid to the producers. The producers transport biomass from the fields to SSLs with their own equipment. The SSL locations and the allocation of production fields to SSLs for the testbed used in this study were established by two previous studies, namely, Resop et al. (2011) and Judd et al. (2012). The flow of biomass into each SSL during each time period is assumed known. This in-flow of biomass depends on the harvest schedule. The effects of different harvest schedules have been empirically studied by Grisso et al. (2013).

Once the biomass is in place, it must to be loaded onto highway-hauling vehicles (tractor-trailers), and transported to the bio-energy plant. The load-out equipment-set is simply a telehandler and a support service truck. A single person operates the telehandler to load

bales into racks during a 10-hour workday. The concept of a rack system for hauling biomass was introduced by Cundiff et al. (2004). A rack can accommodate two levels of 10 bales, and these bales, once loaded in a rack, become a single handling unit for all subsequent operations. A tractor-trailer truck, using a unique low-deck trailer design, is used for highway hauling of the racks. The current concept calls for two trailers in tandem with one rack on each trailer. The total load of two 20-bale racks is 16.3 Mg (15% moisture content).

The filled-racks are transported to the bio-energy plant by truck-tractors pulling the tandem trailers. This configuration is hereafter referred to as a tractor-trailer. During every time period, a number of tractor-trailers is allocated to the SSLs being unloaded to perform out-and-back trips to the bio-energy plant. The number of out-and-back trips that a tractor-trailer can perform during each time period depends on the distance between the SSL to which it is assigned and the bio-energy plant. The tractor-trailers arrive at SSLs with empty racks, and the trailers are unhooked. These racks are then loaded with bales using a telehandler. Meanwhile, the loaded racks are towed to the bio-energy plant. Because of the decoupling of the rack loading operation from the hauling operation, the tractor-trailers do not have to wait to be loaded. Ideally, a tractor-trailer with empty racks will arrive just as the telehandler finishes loading the two empty racks delivered during the previous trip. Delays will always be encountered in a production operation, but an efficient control of all the assets will minimize these delays. The aim of our approach is to better manage these operations.

A supporting truck and trailer of 6-Mg capacity is assumed to be stationed at the bio-energy plant, and this vehicle is used to transport a telehandler between SSLs. At the end of each time period, a decision is made whether to keep the load-out equipment at the same SSL or move it to a different one, a move that is executed with the help of the truck and trailer brought from the bio-energy plant. The truck and trailer is returned to the plant after transporting the telehandler to a new SSL.

At the bio-energy plant, operating 24/7, a homogeneous industrial-scale system loads/unloads tractor-trailers. Full racks are removed with a forklift and replaced with empty racks (a procedure that minimizes unload time), and then trailers with empty racks are towed to an SSL.

2.2.1 Problem statement and literature review

The BL-SCDP can be concisely defined as follows: Given a set F of pre-located SSLs, incoming biomass during each time period t at each SSL i, A_i^t, and a planning horizon of length T, determine optimal number of load-out equipment sets (telehandlers), number of tractor-trailers, storage capacity at each SSL, shipping of biomass from SSLs to the bio-energy plant, inventory levels of biomass at the SSLs, routes of load-out equipment sets, and routes of vehicles during each time period, so as to minimize the total cost incurred due to: (a) the ownership cost of load-out equipment sets and tractor-trailers, (b) mobilization of the load-out equipment, and (c) total cost of shipping biomass from the SSLs to the bio-energy plant.

The time-expanded network in Figure 2.2 illustrates key features of the problem. It depicts four pre-located SSLs (F), the bio-energy plant (denoted by 0), and the planning horizon (T) of 6 periods. During each time period, t, D^t tons of biomass must be delivered from the SSLs to the plant. A_i^t tons of biomass is assumed to be flowing from the fields into SSL i during period t. The decisions to be made during each time period, t, are: $(s_i^t, y_i^t, v_i^t, z_i^t)$, where s_i^t indicates the tons of biomass shipped to the bio-energy plant from SSL i, y_i^t denotes the inventory in SSL i at the start of time period t, v_i^t captures the number of tractor-trailers assigned to perform back-and-forth trips from SSL i to the bio-energy plant, and z_i^t represents the number of load-out equipment sets assigned to load round bales into the racks at SSL i. Additionally, the routes for mobile equipment must be determined. Let x_{ij}^t denote the number of load-out equipment sets traveling from SSL i to SSL j at the end of period t, and

thus, we have, $x^0_{01} = 1$, $x^0_{03} = 1$, $x^1_{12} = 1$, $x^1_{34} = 1$, $x^2_{22} = 1$, $x^2_{43} = 1$, $x^3_{21} = 1$,$x^3_{33} = 1$, $x^4_{12} = 1$, $x^4_{32} = 1$, $x^5_{21} = 1$, $x^5_{24} = 1$, $x^6_{10} = 1$, and $x^6_{40} = 1$. In this example, two load-out equipment sets denoted by E1 (solid line) and E2 (dotted line) start loading-out SSL 1 and SSL 3 in period 1 and then they are moved to SSL2 and SSL 4, respectively. The routes for E1 and E2 for the planning horizon are E1: 1(1)- 2(2)- 2(3)-1(4)-2(5)-1(6)) and E2: 3(1) -4(2) -3(3)-3(4)-2(5)-4(6), where the notation *i(t)* indicates a load-out equipment set unloading SSL i during time t. Thus, for each equipment set, a schedule has to be generated that stipulates which SSLs to visit during the planing horizon.

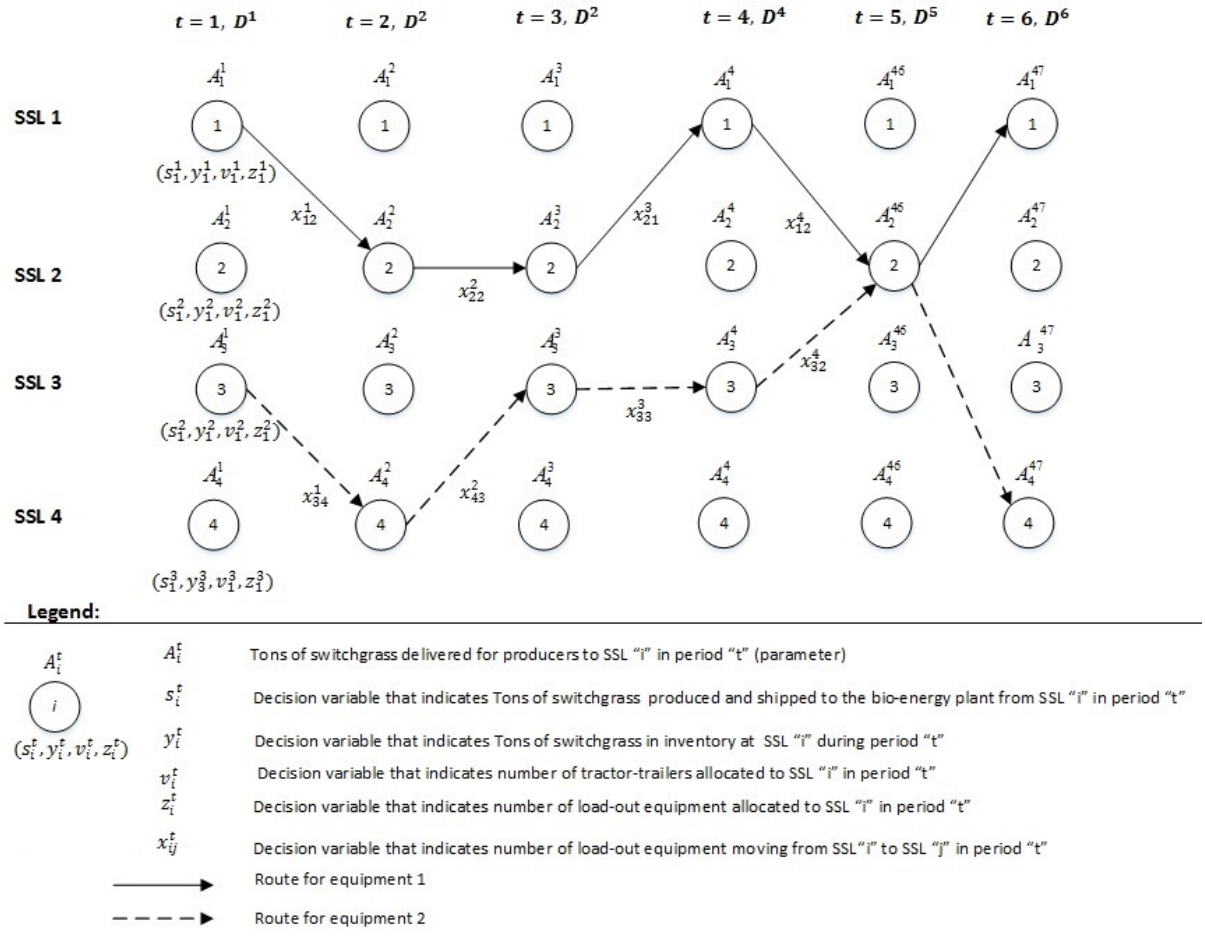

Figure 2.2: Features of the problem in a time-expanded network.

The BL-SCDP is a new and challenging problem that lies at the interface of three areas: single-item capacitated lot-sizing problem with multiple plants or facilities (CLSP-MF), time-

dependent selective high-multiplicity multiple travelling salesmen problem (TDS-HMmTSP), and biomass logistics. The CLSP-MF aspect of the BL-SCDP is as follows (see Figure 2.2): the single item corresponds to bales, lot sizing corresponds to the determination of the amount of biomass to be shipped from each SSL (s_i^t), and multiple facilities are represented by SSLs, each independent of the others, (no transfers of biomass is permitted among SSLs, and each SSL has its own inventory). Capacities or resources are: (1) the rate at which an equipment can load-out biomass, (2) the rate at which a tractor-trailer can haul the biomass to the plant, and (3) the availability of biomass at each SSL. Additionally, the availability of bales from period to period can vary, and the set-up cost corresponds to the mobilization cost of equipment. Zangwill (1966) introduced the idea of multiple facilities in lot-sizing, and also studied facility configurations including parallel resources. He assumed each facility to satisfy its own market requirement, and this aspect is different from the case on hand. Sung (1986) addressed an uncapacitated single-item lot sizing problem with parallel facility, and has used a dynamic programming approach for its solution. Sambasivan and Yahya (2005) have presented a multiple-item capacitated lot-sizing problem with multiple facilities and the possibility of transferring items between facilities, and have proposed a Lagrange-based heuristic approach. Their problem is also different from ours. They consider multiple items, but assume a known market requirement for each item at each facility.

The routing of equipment can be viewed as a time-dependent, selective, high-multiplicity, multiple travelling salesmen problem (TDS-HMmTSP), defined as follows. Given T time periods, N cities, and a distance matrix $[c_{ij}^t]$, find m, minimum routes that start and end at the base city after visiting a minimum number of cities during each time period t, where each city can be visited multiple times, and a city can be served by multiple salesmen at the same time. In the BL-SCDP, time-dependence arises because of the discrete planning horizon (T), and the cities and salesmen are represented by SSLs and equipment sets, respectively. Note that the term selective is based on the premise that not all SSLs need to be visited over the planning horizon, and during each time period only a subset of SSLs may be visited to

ensure the given demand of biomass at the bio-energy plant. High-multiplicity arises because an SSL can be visited multiple times, and multiple salesmen depict the use of several load-out equipment sets to load-out SSLs. Furthermore, an SSL can be served at the same time by multiple load-out equipment sets. The TDS-HMmTSP feature of the problem is better illustrated in Figure 2.3. The routes for equipments E1 and E2 (previously shown in Figure 2.2) are presented differently here to highlight the high-multiplicity feature. $E1$ departs from the bio-energy plant (P) at the end of time 0 to load-out bales at SSL1. Then, at the end of period 1, it is transported to SSL 2. At the end of period 2, $E1$ stays at the same location, i.e, SSL 2. At the end of period 3, $E1$ is transported to SSL 1 again. Here, E1 visits, SSLs 1 and 2 three times each, meanwhile, E2 visits SSLs 3, 4 and 2 respectively three, two and one times. In this case, all SSLs are visited, but in general this need not to be the case.

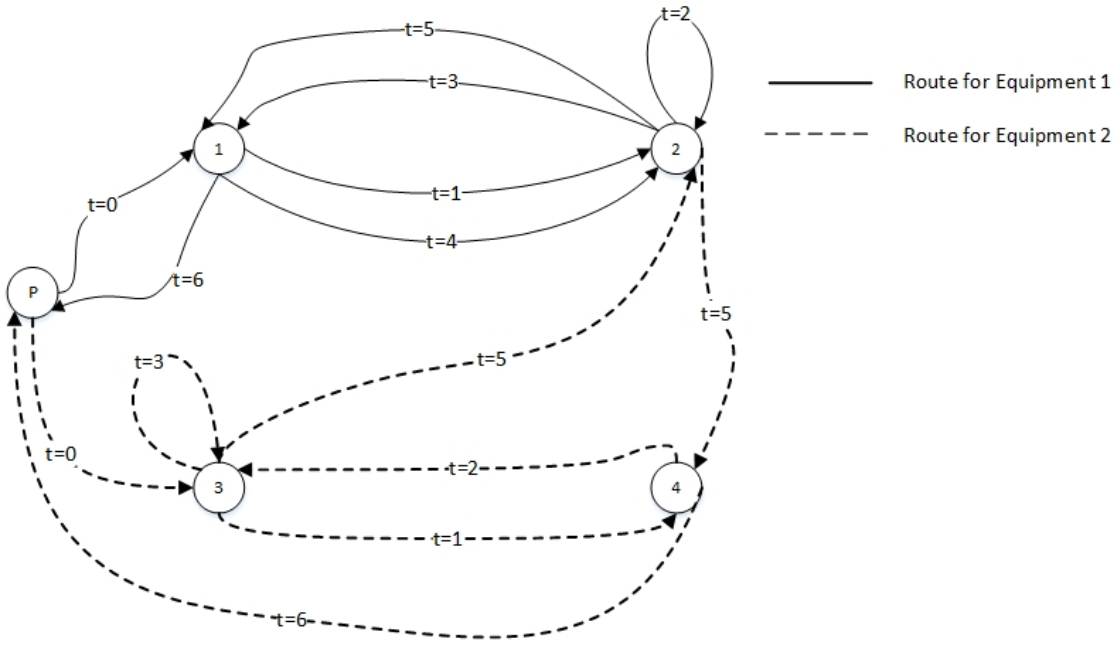

Figure 2.3: The high-multiplicity multiple traveling salesmen problem feature of the BL-SCDP.

To the best of our knowledge, the TDS-HMmTSP has not been studied in the literature; however, its individual features have been addressed. For example, the time-dependent

feature has been addressed in the context of traveling salesman problem, where the cost associated with an arc depends of its position in the tour with respect to the base city (see Picard and Queyranne (1978), Fox et al. (1980) and Gouveia and Voß (1995)). The selection feature has been studied, albeit under different names: selective TSP (Laporte and Martello (1990); Gendreau et al. (1998)), TSP with profits (Feillet et al. (2005)), and Orienteering Problem (Vansteenwegen et al. (2011)). For the TSP, selection implies that it is not necessary to visit all the cities, and usually, there is a profit for collection that is associated with each city. The high multiplicity asymmetric travelling salesman problem (HMATSP) has been addressed by Grigoriev and van de Klundert (2006) (who proposed a exponential-size formulation for the problem), and later by Sarin et al. (2011) (who developed a polynomial-length formulation for the HMATSP that is effective to use). The readers interested in the m-ATSP are referred to two comprehensive reviews of the problem by Bektaş (2006) and Sarin et al. (2014a)

In the Biomass Logistic literature, our problem can be categorized as a multi-period problem. Although there exists a rich literature in this area, we present a more comprehensive model that incorporates several practical strategic and tactical features. Multi-period formulations that minimize the total annual cost given that the demand at the plant for each period must be meet, has been addressed by Cundiff et al. (1997), Eksioglu et al. (2009), Huang et al. (2010), Zhang et al. (2013), Lin et al. (2014), and Xie et al. (2014). Multi-period models with profit maximization instead of cost minimization have been reported by Tembo et al. (2003), An et al. (2011), and Zhu and Yao (2011), Zhu et al. (2011). However, none of these studies has included in its model operations of load-out equipment sets. Only Judd et al. (2012) have included this feature in their study, however, they consider only a single-period problem.

2.3 Model formulation and valid inequalities

We formulate the BL-SCDP as a mixed integer programming model. Consider the following notation.

Sets:

F	:	Set of SSLs.
F^0	:	Set of SSLs including the bio-energy plant, which is denoted by 0 $(0 \cup F)$.
T	:	Length of the planning horizon $(1, ..., T)$.
T^0	:	Length of the planning horizon including time 0, $(0, 1, ..., T)$.
T^+	:	Length of the planning horizon including time T+1, $(1, ..., T+1)$.

Parameters:

$\hat{c}_i^t$	:	Shipping cost from SSL i to the bio-energy plant in period t (\$ per Mg), $i \in F$.
c_e	:	Equipment purchasing cost (\$ per unit).
c_v	:	Tractor-trailer purchasing cost (\$ per unit).
c_{ij}^t	:	Cost to transport an equipment from SSL i to SSL j in period t (\$ per mile), $\forall i \in F$, $\forall j \in F$.
A_i^t	:	Amount of biomass available at SSL i during time period t (Mg), $\forall i \in F, \quad \forall t = 0, ..., T$.
N_i	:	Number of out-and-back trips a vehicle can perform during a time period if allocated to SSL i, $\forall i \in F$.
α	:	Fraction of biomass inventory lost.
Q	:	Tractor-trailer capacity (Mg).
U	:	Maximum load-out rate for each equipment set (Mg per time unit).
P^t	:	Plant requirement during time period t (Mg), $\forall t = 1, ..., T$.

E : Maximum number of equipment sets to allocate to an SSL during a time period.

Decision Variables:

s_i^t : Amount of biomass shipped from SSL i to the bio-refinery during period t, $\forall i \in F, \quad \forall t = 1, ..., T$.

y_i^t : Inventory of biomass at the end of time period t at SSL i. $\forall i \in F, \quad \forall t = 0, ..., T$.

v_i^t : Number of tractor-trailers assigned to SSL i during time period t, $\forall i \in F, \quad \forall t = 1, ..., T$.

z_i^t : Number of equipment sets assigned to SSL i during time period t, $\forall i \in F, \quad \forall t = 1, ..., T$.

K : Number of load-out equipment sets purchased.

V : Number of tractor-trailers purchased.

x_{ij}^t : Number of equipment sets traversing from SSL i to SSL j at the end of time period t. $\forall i \in F^0, \quad \forall j \in F^0, \forall t = 0, ..., T$.

Model BL-SCDP:

$$\text{Minimize} \quad c_e K + c_v V + \sum_{t=1}^{T+1} \sum_{i \in F^0} \sum_{j \in F^0} c_{ij} x_{ij}^{t-1} + \sum_{t=1}^{T} \sum_{i \in F} \hat{c}_i^t s_i^t \tag{2.1}$$

subject to :

$$\sum_{i \in F} s_i^t \geq P^t, \quad \forall t = 1, ..., T \tag{2.2}$$

$$y_i^t = (1 - \alpha) y_i^{t-1} - s_i^t + A_i^t, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.3}$$

$$y_i^0 = A_i^0, \quad \forall i \in F \tag{2.4}$$

$$s_i^t \leq Q N_i v_i^t, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.5}$$

$$\sum_{i \in F} v_i^t \leq V, \quad \forall t = 1, ..., T \tag{2.6}$$

$$\sum_{i \in F} z_i^t \leq K, \quad \forall t = 1, ..., T \tag{2.7}$$

$$s_i^t = U z_i^t \quad , \forall i \in F, \quad \forall t = 1, ..., T \tag{2.8}$$

$$\sum_{j \in F^0} x_{ji}^{(t-1)} - z_i^t = 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.9}$$

$$\sum_{j \in F^0} x_{ji}^{(t-1)} - \sum_{j \in F^0} x_{ij}^t = 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.10}$$

$$x_{i0}^t = 0, \quad \forall i \in F, \quad \forall t = 0, ..., T-1 \tag{2.11}$$

$$s_i^t \geq 0, \quad \forall i \in F, \quad \forall t = 1, ...T \tag{2.12}$$

$$z_i^t \quad \in \{0, 1, ..., E\}, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.13}$$

$$y_i^t \geq 0 \ , \quad \forall i \in F, \quad \forall t = 0, ..., T \tag{2.14}$$

$$v_i^t \in Z^+, \quad \forall j \in F, \quad \forall t = 1, ..., T \tag{2.15}$$

$$x_{ij}^t \in \{0, 1, ..., E\}, \quad \forall i \in F^0, \quad \forall j \in F^0, \quad \forall t = 0, ..., T \tag{2.16}$$

$$V \in Z^+ \tag{2.17}$$

$$K \in Z^+ \tag{2.18}$$

The objective function (2.1) captures our aim of minimizing the total cost incurred by the purchase of both the load-out equipment sets and tractor-trailers, mobilization of equipment, and shipping of biomass from SSLs to the bio-energy plant. Constraints (2.2) assure that the plant requirement during each time period t is satisfied. Constraints (2.3) capture the inventory balance during each time period. Constraints (2.4) correspond to the initial inventory at SSL i in period 0. Constraints (2.5) ensure that the amount shipped from SSL i to the plant in each time period t is at most equal to the number of tractor-trailers (v_j^t) allocated to this SSL times their capacity (Q) and the number of trips (N_i). Constraints (2.6) and (2.7) capture, respectively, the maximum number of tractor-trailers and load-out equipment sets required. Constraints (2.8) assure that bales can be shipped from SSL i to the bio-energy plant only if there is a load-out equipment allocated to that SSL. Furthermore, it makes sure that a load-out equipment is allocated to an SSL during time period t if and only if it is fully utilized during each time unit- a requirement imposed in practice. Constraints (2.9)

enforce mobilization of required number of equipment sets from SSL $j \in F^0$ to SSL $i \in F$ at the end of a time period t. Note that we permit an equipment set to stay at an SSL if needed. Constraints (2.10) are the standard flow conservation constraints for equipment sets. Constraints (2.11) enforce an equipment set to stay at an SSL and not move to the bio-energy plant until the last time period. Constraints (2.12)- (2.18) define the domains of the variables.

2.3.1 Valid inequalities for the BL-SCDP

Next, we present some valid inequalities for the BL-SCDP

Proposition 2.3.1. *The following inequalities are valid for the BL-SCDP*

$$\sum_{j \in F^0} \sum_{i \in F} x_{ji}^{t-1} \geq \max\{ \quad \lceil P^t/U \rceil \quad \}, \quad \forall t = 1, ..., T \tag{2.19}$$

$$\sum_{i \in F} z_i^t \geq \max\{ \quad \lceil P^t/U \rceil \quad \}, \quad \forall t = 1, ..., T \tag{2.20}$$

$$v_i^t \leq (U z_i^t)/Q \quad \forall i \in F \quad , \forall t = 1, ..., T \tag{2.21}$$

Proof. Inequalities (2.19) enforce the number of load-out equipment sets during each time period to be at least equal to the minimum number required to satisfy the demand during every period because otherwise the demand at the plant will not be met during at least one period.

Inequalities (2.20) follow by adding constraints (2.9) over $f \in F$, and then using (2.19).

Regarding inequalities (2.21), $(U z_i^t)$ indicates the total amount of biomass available to transport to the bio-energy plant from SSL i in period t. Dividing this amount by Q (tractor-trailer capacity), gives an upper bound on the number of tractor-trailers required to transport this biomass to the bio-energy plant. Note that if $z_i^t = 0$, then $v_i^t = 0$ as well. □

2.4 Branch-and-price approach

We develop a column generation (Desaulniers et al. (2005)) approach that exploits the fact that only expressions (2.9) are the complicating constraints that link the capacitated lot-sizing problem for multiple facilities (CLSP-M) (biomass supply), (2.2)-(2.8), with the TDS-HMmTSP problem (equipment mobilization) (2.10)- (2.11). Therefore, we can decompose the problem into two parts: master problem, and subproblem. The master problem comprises of constraints (2.9)-(2.11), and the subproblem consists of constraint sets (2.2)-(2.8). The domain of the variables are incorporated in these problems accordingly. We present these problems next, and describe strategies to implement our overall branch-and-price approach (B&P).

First, we introduce additional notation required for the formulation of the master and sub-problem.

Notation:

- S : Set of extreme points corresponding to constraints (2.2)- (2.8).
- λ_s : Binary variables indicating if column s is selected. $\forall s \in S$.
- c_s : Cost of column s, $s \in S$.
- (z_i^t) : Value indicating number of equipment sets visiting SSL i during period t.
- μ_i^t : Dual variables associated with constraints (2.23), $\forall i \in F$, $\forall t = 1, ..., T$.
- π : Dual variable associated with constraint (2.26).

Then, the master problem is given by:

$$\text{MP: Minimize} \quad \sum_{s \in S} c_s \lambda_s + \sum_{t=1}^{T+1} \sum_{i \in F^0} \sum_{j \in F^0} c_{ij} x_{ij}^{t-1} \tag{2.22}$$

subject to :

$$\sum_{j \in F^0} x_{ji}^{t-1} - \sum_{s \in S} (z_i^t) \lambda_s = 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \qquad (\mu_i^t) \tag{2.23}$$

$$\sum_{j\in F^0} x_{ji}^{t-1} - \sum_{j\in F^0} x_{ij}^{t} = 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.24}$$

$$\sum_{j\in F^0}\sum_{i\in F^0} x_{ji}^{t-1} \geq \max\{ \quad \lceil P^t/U \rceil \quad \}, \quad \forall t = 1, ..., T \tag{2.25}$$

$$\sum_{s\in S} \lambda_s = 1 \qquad (\pi) \tag{2.26}$$

$$x_{i0}^{t} = 0, \quad \forall i \in F, \quad \forall t = 0, ..., T \tag{2.27}$$

$$x_{ij}^{t} \in \{0, 1, ..., E\}, \quad \forall i \in F^0, \quad \forall j \in F^0, \quad \forall t = 0, ..., T \tag{2.28}$$

$$\lambda_s \in \{0, 1\} \quad \forall s \in S \tag{2.29}$$

We solve the master problem by relaxing the integrability requirements of x_{ij}^t and λ_s, and call it as a relaxed master problem. Note that we add the valid inequality (2.19) presented above in (2.25) to strengthen the linear programming relaxation of the master problem.

The reduced cost is given by:

$$\bar{c} = c_s - (\sum_{t=1}^{T}\sum_{i\in F} -\mu_i^t z_i^t) - \pi = c_e K + c_v V + \sum_{t=1}^{T}\sum_{i\in F} c_i^t s_i^t + \sum_{t=1}^{T}\sum_{i\in F} \mu_i^t z_j^t - \pi \tag{2.30}$$

Then, the subproblem is formulated as follows:

SP: Minimize (2.30)

subject to :

$$\sum_{i\in F} s_i^t \geq P^t, \quad \forall t = 1, ..., T \tag{2.31}$$

$$y_i^t = (1-\alpha) y_i^{t-1} - s_i^t + A_i^t, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.32}$$

$$y_i^0 = A_i^0, \quad \forall i \in F \tag{2.33}$$

$$s_i^t = U z_i^t, \quad \forall i \in F \quad , \forall t = 1, ..., T \tag{2.34}$$

$$s_i^t \leq Q N_i v_i^t, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.35}$$

$$v_i^t \leq (U z_i^t)/Q, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.36}$$

$$\sum_{i\in F} z_i^t \geq \max\{ \quad \lceil P^t/U \rceil \quad \}, \quad \forall t = 1, ..., T \tag{2.37}$$

$$\sum_{i \in F} z_i^t \leq K, \quad \forall t = 1, ..., T \tag{2.38}$$

$$\sum_{i \in F} v_i^t \leq V, \quad \forall t = 1, ..., T \tag{2.39}$$

$$s_i^t \geq 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.40}$$

$$y_i^t \geq 0, \quad \forall i \in F, \quad \forall t = 0, ..., T \tag{2.41}$$

$$v_i^t \in Z^+, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.42}$$

$$z_i^t \in \{0, 1, ..., E\}, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.43}$$

$$V \in Z^+ \tag{2.44}$$

$$K \in Z^+ \tag{2.45}$$

The subproblem is solved as a mixed-integer programming problem. Also, we have added the valid inequalities (2.20) and (2.21) in SP as constraints (2.36) and (2.37) to strengthen the formulation.

2.4.1 Implementation strategies

Branching

The columns generated by the subproblem are added to the restricted master problem. At the termination of this column generation procedure, the solution need not be integer. That is, x_{ij}^t may have continuous values. The solution so obtained is called the node "0" solution. We branch on x_{ij}^t variables in order to obtain an integer optimal solution. The column generation procedure is repeated at each node resulting from branching.

Let $x_{ij}^t = a$ (a fractional value). We branch on x_{ij}^t. On one side of the tree, we enforce in the master problem $x_{ij}^t \geq \lceil a \rceil$, and add additional requirements in the subproblem depending on the value of t. If $t \neq 0$ and $t \neq T$, then we enforce in the subproblem both $z_i^t \geq \lceil a \rceil$ and $z_j^{t+1} \geq \lceil a \rceil$, i.e, SSL i and j have to be visited at least $\lceil a \rceil$ times in periods t and $t+1$, respectively, since at least $\lceil a \rceil$ load-out equipment sets are traversing arc (i, j) at the end

of period t. If $t = 0$, i.e, $x^0_{0j} \geq \lceil a \rceil$, we enforce only $z^1_j \geq \lceil a \rceil$. If $t = T$, i.e, $x^T_{i0} \geq \lceil a \rceil$, we enforce $z^T_i \geq \lceil a \rceil$.

On the other branch of the tree, we enforce in the master problem $x^t_{ij} \leq \lfloor a \rfloor$, and include the following additional requirements in the subproblem depending on the value of t. If $t = 0$, i.e, $x^0_{0j} \leq \lfloor a \rfloor$, we enforce in the subproblem $z^1_j \leq \lfloor a \rfloor$. If $t = 1$, i.e, $x^1_{ij} \leq \lfloor a \rfloor$, we enforce $z^1_i \leq \lfloor a \rfloor$ because $x^0_{0i} \leq \lfloor a \rfloor$. If $t = T$, i.e, $x^T_{i0} \leq \lfloor a \rfloor$, we enforce $z^T_i \leq \lfloor a \rfloor$ because all equipment sets must return to "0". However, for $t \neq 0$, $t \neq 1$, and $t \neq T$, neither $z^{t+1}_j \leq \lfloor a \rfloor$ nor $z^t_i \leq \lfloor a \rfloor$ can be enforced since SSL j and i can be visited from another SSL, and thus, even for $x^t_{ij} \leq \lfloor a \rfloor$, we could have z^t_i, $z^{t+1}_j \not\leq \lfloor a \rfloor$.

Furthermore, we have implemented the best bound strategy for node selection, i.e, we choose a node with the smallest linear relaxation value. Once a node is selected, we choose a x^t_{ij} variable with the smallest t and i indexes to branch from.

Generation of an upper bound

We add all the columns generated during the branch-and-price approach to the relaxed master problem. Also, we add these to an auxiliary master program (a copy of the relaxed master problem) that is created to compute an upper bound by imposing integrability on x^t_{ij} and λ_s variables. When the pool of columns added to this auxiliary model reaches a prescribed size given by the parameter *Pool Size*, the auxiliary master program is solved by CPLEX for a prescribed solution time. If an optimal solution is obtained within this allowed time, i,e, the best column is selected, then all the non-optimal columns are dropped from this auxiliary model (leaving only the best column in the pool), and the *Pool Size* is set to zero. Otherwise, we keep all the columns in the auxiliary master program, and set *Pool Size* to zero. This process is repeated every time the pool reaches its capacity, and therefore, it provides an upper bound each time.

Our computational experimentation shows that this strategy is very effective for large-sized problem instances, where we are able to generate upper bounds in seconds instead of min-

utes or hours of clock time, a drastic speed-up step contributing to the effectiveness of the proposed B&P approach.

Generation of a lower bound

Usually in column generation, each subproblem is not solved to optimality as it is very time consuming to do so. Instead, heuristic approaches are utilized to speed up the generation of columns. One drawback of this strategy is that it does not lend itself to the generation of a lower bound. Therefore, we use the following strategy to compute a lower bound where the subproblem is not solved to optimality.

First, we introduce additional notation. Let z^*_{lp} be the optimal linear relaxation value of the master problem (2.22)- (2.29), $\bar{z}^k_{lp}$ be the optimal linear relaxation value of the restricted master problem at iteration k, v^k be the optimal reduced cost at iteration k, $\underline{v}^k$ be a lower bound on the optimal reduced cost at iteration k, and z_{lb} be the best lower bound until iteration k. Then, we can determine z_{lb} as follows:

Proposition 2.4.1. *z_{lb} given by*

$$z_{lb} = \min_{k=1,\dots,K} (\bar{z}^k_{lp} + \underline{v}^k) \tag{2.46}$$

is a valid lower bound for the overall problem.

Proof. We have $\bar{z}^k_{lp} + v^k \leq z^*_{lp}$, and since $\underline{v}^k \leq \bar{v}^k$, it follows that $\bar{z}^k_{lp} + \underline{v}^k \leq \bar{z}^k_{lp} + v^k \leq z^*_{lp}$. Therefore, $\min_{k=1,\dots,K}(\bar{z}^k_{lp} + \underline{v}^k) \leq \bar{z}^k_{lp} + \bar{v}^k \leq z^*_{lp}$. □

Note that $\underline{v}^k$ is at hand at each iteration since the subproblem is solved directly using CPLEX, and it can be used to terminate the B&P procedure when a prescribed optimality gap is reached. The effectiveness of the proposed lower bound generation strategy is highlighted in Section 2.5, where it provides tighter lower bound than those obtained by directly solving the BL-SCDP model using CPLEX.

Initial set of columns

In order to improve the convergence of the column generation approach, we determine initial solutions to the large-scale integrated model that serve as initial columns for the master problem. We generate these by solving a sub-model given by constraints (2.2)- (2.8), and the modified objective function given by:

$$c_e K + c_v V + \sum_{t=1}^{T} \sum_{i \in F} c_i^t s_i^t \tag{2.47}$$

Note that this sub-model corresponds to the lot-sizing aspect of the problem, and thus, does not considerer the mobilization component (TDS-HMmTSP). We solve this sub-model as mixed-integer program using CPLEX for a prescribed time or until a desired optimality gap is reached, and generate several solutions that form the initial columns for the master problem.

Dual stabilization

A dual stabilization technique (du Merle et al. (1999)) aims to improve the convergence of the column generation method, which can be often very slow when solving degenerate restricted master problems. The MP is highly degenerate, thereby causing large oscillations of the dual variable values between iterations. To mitigate these undesired effects and improve the convergence of our B&P approach, we have used a dual stabilization technique which is described next. First, we need the following additional notation:

k	:	Number of iterations performed thus far.
$w_i^{t+}(w_i^{t-})$	:	Surplus (Slack) variable, $\forall i \in F \quad \forall t = 1, ..., T$.
μ_i^{tk}	:	Dual price associated with constraints (2.49), $\forall i \in F$, $\forall t = 1, ..., T$, at the k^{th} iteration.

$\bar{\mu}_{i+}^{tk}(\bar{\mu}_{i-}^{tk})$: Bound parameter (upper and lower bound, non-positive and non-negative, respectively) for the dual prices μ_i^{tk}, $\forall i \in F \;\; \forall t = 1, ..., T$, at the k^{th} iteration.

$$P(x,k): \text{Minimize} \quad \sum_{s\in S} c_s\lambda_s + \sum_{t=1}^{T+1}\sum_{i\in F^0}\sum_{j\in F^0} c_{ij}x_{ij}^{(t-1)} + \sum_{t=1}^{T}\sum_{i\in F}(\bar{\mu}_{i+}^{tk})w_i^{t+}$$

$$-\sum_{t=1}^{T}\sum_{i\in F}(\bar{\mu}_{i-}^{tk})w_i^{t-} \tag{2.48}$$

$$\sum_{j\in F^0} x_{ji}^{(t-1)} - \sum_{s\in S}(z_i^t)\lambda_s + w_i^{t+} - w_i^{t-} = 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \quad (\mu_i^{tk}) \tag{2.49}$$

$$\sum_{j\in F^0} x_{ji}^{(t-1)} - \sum_{j\in F^0} x_{ij}^{t} = 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.50}$$

$$\sum_{j\in F^0}\sum_{i\in F^0} x_{ji}^{t-1} \geq \max\{ \quad \lceil P^t/U \rceil \quad \}\rceil, \quad \forall t = 1, ..., T \tag{2.51}$$

$$\sum_{s\in S}\lambda_s = 1 \qquad (\pi) \tag{2.52}$$

$$x_{i0}^{t} = 0, \quad \forall i \in F, \quad \forall t = 0, ..., T-1 \tag{2.53}$$

$$x_{ij}^{t} \in \{0, 1, ..., E\}, \quad \forall i \in F^0, \quad \forall j \in F^0, \quad \forall t = 0, ..., T \tag{2.54}$$

$$\lambda_s \in \{0, 1\}, \quad \forall s \in S \tag{2.55}$$

$$w_i^{t+} \geq 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.56}$$

$$w_i^{t-} \geq 0, \quad \forall i \in F, \quad \forall t = 1, ..., T \tag{2.57}$$

Dual variables (μ_i^{tk}) are subject to the following constraints

$$\bar{\mu}_{i-}^{tk} \leq \mu_i^{tk} \leq \bar{\mu}_{i+}^{tk}, \quad \forall i \in F, \quad \forall t = 1, ..., T. \tag{2.58}$$

The values of $\bar{\mu}_{i-}^{tk}$ and $\bar{\mu}_{i+}^{tk}$ are updated as follows, where ϵ is a prescribed perturbation

amount.

$$\bar{\mu}_{i+}^{tk} = \bar{\mu}_{i+}^{t(k-1)} + \epsilon \tag{2.59}$$

$$\bar{\mu}_{i-}^{tk} = \bar{\mu}_{i-}^{t(k-1)} - \epsilon \tag{2.60}$$

We assume a starting value of $\epsilon = 0.005$. At the root node, we start by centering the duals at the origin, and solve the master problem for *ItNum* number of iterations. Once this number has been reached, we increase ϵ by a factor of 10. Meanwhile, at any node during the branch-and-price procedure, the duals are centered at the dual values corresponding to the predecessor of the current node, and then processed as before.

Column management

We avoid overpopulating the master problem by dropping unnecessary columns. Thus, the columns that have not been selected to be in any basis while solving the master problem for a prescribed number of iterations are dropped from the master problem.

Initial solution for the subproblem

An initial solution is provided to the subproblem to reduce computational time. Let k be the current iteration, and s_*^k be the optimal or best-known solution to the subproblem solved in iteration k. Thus, for $k \geq 2$, a warm start solution (or initial solution) corresponding to s_*^{k-1} is provided to drastically reduce the computational time. For the large-sized instances addressed in this chapter, this strategy enables us, each time the subproblem is solved, to generate columns with negative reduced cost in seconds instead of a couple minutes or more of wall clock time.

Generation of multiple columns

Each time the subproblem is invoked, several columns with negative reduced costs are returned to the master problem in order to improve its convergence.

Termination criteria for the subproblem

The subproblem optimization process is monitored at all times, and as soon as a column with a negative reduced cost is identified, it is returned to the master problem. The procedure terminates when a column with negative reduced cost cannot be obtained.

Termination criteria for the master problem

The generation of columns for the master problem continues until one of the following three criteria is met: (i) no column with negative reduced cost can be identified, (ii) global time limit is reached, or (iii) the linear relaxation gap computed as $(\bar{z}^k_{lp} - z_{lb})/\bar{z}_{lp}$ is less than a prescribed value defined a priory, where $\bar{z}_{lp}$ is the objective value of the relaxed master problem at iteration k and z_{lb} is the best lower bound of the master problem until iteration k.

2.5 Computational results

2.5.1 Case study

The dataset utilized to illustrate the use of the proposed methodology consists of the SSLs as described by Resop et al. (2011). A geographical information system was utilized to identify potential fields for the production of switchgrass. A total of 199 SSL(s) are distributed over a 48-km radius around Gretna, VA. (See Figure 2.4). They were divided into 5 sections with each section having 48, 23, 39, 32, and 57 SSLs, respectively, and having approximately the same biomass production (Mg). Note that a production field was assigned to a unique SSL, but an SSL might provide storage to several fields.

The flow of biomass into each SSL during each week is assumed known, and it depends on the harvest schedule. Here, in-flow rates into SSLs during each week is based on an expected harvest schedule, based on typical weather. It is as follows: Month 1 (1%), Month

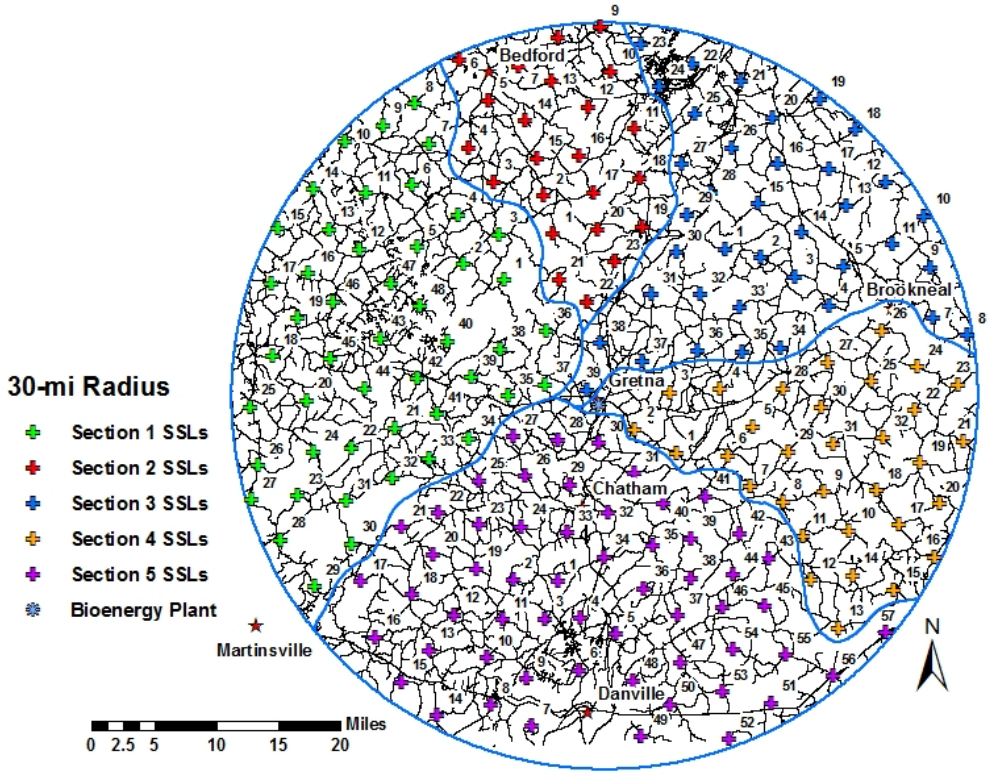

Figure 2.4: Two-dimensional picture displaying SSL(s) and bio-energy plant location at Gretna, VA. (Source: Liu et al. (2013), InTech, Open Access, 2013)

2 (6%), Month 3 (13%), Month 4 (21%), Month 5 (17%), Month 6 (13%), Month 7 (9%), Month 8 (9%), Month 9 (10%), Month 10 (0%), Month 11 (0%), and Month 12 (0%). The percentages given are the percent of total annual harvest from production fields assigned to a given SSL that is harvested in the given month. Material flows from the production fields into each SSL during each month except for the months of April (month 10), May (month 11), and June (month 12) because the crop is growing during these months. Furthermore, we assumed a planing horizon of 48 periods (weeks), and thus, each month contains 4 weeks, and the biomass-flow into each SSL is equally distributed over the weeks of a month.

The dataset consists of five sections (I, II, III, IV and V), and we generate four groups of problems by aggregating SSL sections. Thus, Group 1 consists of section I and II, Group 2 includes Sections I, II, and III, Group 3 incorporates Sections I, II, III, and IV, and finally, Group 4 combines Sections I, II, III, IV, and V. Furthermore, for each group, we

generate four different instances by modifying input parameters which include number of periods (T), number of satellite storage locations (#SSL), bio-energy plant requirement in each period (P^t), and load-out equipment capacity in each time period (U). A total of 16 problem instances are generated, and details of each instance are displayed in Table 2.1. Note that the plant requirement and equipment capacity are varied arbitrarily for problems 1-12 in order to generate a feasible solution given the input parameters. For instances 13-16, equipment capacity is kept fixed (605 Mg per week), which is the actual rate at which a telehandler can load-out bales, but the plant requirement is varied at four different levels.

Table 2.1: Problem instances

Instance	**Group**	**T**	**# SSL(s)**	**P (Mg)**	**U (Mg)**
1	1	48	71	2000	300
2	1	48	71	2000	605
3	1	48	71	2500	300
4	1	48	71	3000	300
5	2	48	110	3000	605
6	2	48	110	3500	605
7	2	48	110	4000	300
8	2	48	110	4000	605
9	3	48	142	4000	605
10	3	48	142	4500	605
11	3	48	142	5000	605
12	3	48	142	5250	605
13	4	48	199	5445	605
14	4	48	199	6050	605
15	4	48	199	6500	605
16	4	48	199	6655	605

The mobilization cost is defined as the cost to move load-out equipment, specifically the telehandler, from one SSL to the next. The following rules were used to estimate mobilization cost: (1). If the next SSL is not more than 8 km away, the telehandler will be driven to the next SSL. Operational cost of the telehandler is \$39.50/h, and the travel time is based on the road speed of 27 km/h. (2) The cost incurred for the truck and trailer to haul the telehandler is \$ 125/h, which includes the labor cost for the truck operator. Total mobilization time

consists of the time required to travel from the bio-energy plant to the current SSL, time to load the telehandler, travel time to the next SSL, time to unload the telehandler, and the travel time back to the bio-energy plant. Moreover, other necessary information is as follows: (a) travel time to an SSL is based on the average road speed of 70 km/h, (b) load time is 20 minutes, (c) travel time with load is based on an average road speed of 65 km/h, (d) unload time is 20 minutes, (e) return travel to bio-energy plant is based on an average road speed of 70 km /h, and (f) cost for the telehandler operator is \$20/h x total mobilization time (There is no additional operating cost for the telehandler since it is not operating during the move). Finally, note that, if a telehander is kept at the same location, then there is no cost incurred.

Other parameters used in the model are as follows. Cost of purchasing load-out equipment and tractor-trailers are $\$94,000$ and $\$50,000$ respectively. As mentioned above, the number of periods corresponds to 48 weeks, and the hauling cost is $0.1381 * d_i + 1.6667$ (\$/Mg) (Judd et al. (2012)), where d_i is the distance (km) from SSL i to the bio-energy plant.

Next, we present results of the application of the proposed branch-and-price procedure to the above described problem instances inspired by a real-life case study. The model BL-SCDP was solved directly using CPLEX 12.5.1 with OPL with default parameters, and the proposed branch-and-price procedure was coded using Microsoft Visual Studio Professional 2012. Both of these procedures were solved on an Intel Xeon E5-2687W 3.5GHz (32 thread) computer with 32 GB of RAM running Windows 7, and the maximum solution time was set to 7,200 seconds (wall clock time). Furthermore, the master problem was solved using concurrent LP solvers (this setting was found to outperform primal simplex and dual simplex as LP solver), where different solvers run in parallel until one reaches the optimal solution, and the subproblem was solved as a mixed-integer program using CPLEX as a solver using default parameters except for the parameter denoted by CPX_PARAM_SOLNPOOLINTENSITY set to a value of 3 (aggressive) in order to generate a large number of solutions.

Other parameters in the proposed B&P are as follows: *Pool Size* and *ItNum* were set to 50 and 5, respectively. The generation of columns at any node including the root node is aborted as soon as the linear programming relaxation gap described in the previous section is under 3% and all artificial variables from the dual stabilization (w_i^{t+} and w_i^{t-}) are dropped from the MP. Also, columns are dropped from the MP after not being selected to be in any basis for 100 iterations.

2.5.2 Integer solution results

Table 2.2 presents a comparison between direct solution of model BL-SCDP by CPLEX and its solution by the proposed branch-and-price (B&P) method. The first two columns indicate the instance number and the best lower bound (Z_{lb}) obtained by either B&P or CPLEX (those obtained by B&P are indicated with BP as superscript). Columns 3-5, respectively, indicate the integer solution (Z_{IP}), integer optimality gap (Opt.Gap) computed as $(Z_{IP} - Z_{lb})/Z_{IP}$), and solution time corresponding to the root node. Columns 6-8 indicate number of nodes explored (Nodes), integer solution (z_{ip}), and the integer optimality gap (Opt.Gap) obtained by the proposed (B&P) approach. Columns 9-10 present the results obtained by the direct solution of model BL-SCDP by CPLEX. Note that, our approach for generating a lower bound (described in Section 3) is quite effective and leads to better lower bounds values in 10 out of 16 cases. With respect to integer solutions, the proposed B&P method outperforms CPLEX in all the instances. CPLEX is unable to solve 6 out of 16 instances because of either excessive memory requirements or reaching the maximum allowed time limit. For those instances that CPLEX was able to solve, the average optimality gap for CPLEX and the proposed B&P, respectively, are 10.7% and 3.2%. Furthermore, note that in 9 out of 10 cases for which CPLEX was able to obtain a solution, the solutions obtained at the root node (before branching) are better than the ones obtained by CPLEX. For the one case for which the CPLEX solution was better, the optimality gap value for the CPLEX

solution is 8.3% as compared with 8.5% for the solution obtained at the root node. For those instances for which CPLEX was unable to obtain a solution, the B&P approach is able to generate solutions with an average optimality gap of 4.99%. Furthermore, the average, minimum, and maximum improvement of B&P over the root node solution are 1.6%, 0.1%, and 6.4%, respectively. The largest improvement occurs for instance 3, where the total cost is decreased by $157,934.

Table 2.2: Comparison between the proposed branch-and-price approach and direct solution of model BL-SCDP by CPLEX

Instances		**Root Node**			**Branch-and-Price**			**CPLEX**	
#	Z_{lb}	Z_{IP}	Opt.Gap	Time	Nodes	Z_{IP}	Opt.Gap	Z_{IP}	Opt.Gap
1	1632856.7	1709683	4.5%	146.93	97	1665282	**1.9%**	1879520	13.1%
2	1549093.0^{BP}	1623545	4.6%	93.83	148	1569218	**1.3%**	1651474	6.2%
3	2224369.4^{BP}	2420308	8.1%	113.18	54	2262374	**1.7%**	2424326	8.2%
4	2475843.0^{BP}	2613846	5.3%	288.12	56	2602884	**4.9%**	- †	-
5	1793816.8	1909273	6.0%	253.98	25	1863110	**3.7%**	2145400	16.4%
6	2318095.6^{BP}	2360959	1.8%	263.96	49	2353049	**1.5%**	2584837	10.3%
7	3395986.7	3588413	5.4%	419.22	5	3574694	**5.0%**	-	-
8	2707026.27^{BP}	2831878	4.4%	221.68	35	2829507	**4.3%**	2941498	8.0%
9	2518157.7^{BP}	2656987	5.2%	530.76	16	2649245	**4.9%**	2966143	15.1%
10	2958392.0^{BP}	3151374	6.1%	346.9	21	3103206	**4.7%**	3424274	13.6%
11	3449003.4^{BP}	3771000	8.5%	237.18	23	3612972	**4.5%**	3761892	8.3%
12	3476700.56^{BP}	3619497	3.9%	348.12	23	3610853	**3.7%**	3761835	7.6%
13	3178919.4	3490422	8.9%	718.3	9	3470173	**8.4%**	-	-
14	3665586.00^{BP}	3955092	7.3%	764.62	10	3937595	**6.9%**	-	-
15	4273208.2	4445528	3.9%	901.99	8	4382855	**2.5%**	-	-
16	4102807.3	4481690	8.5%	750.15	9	4422982	**7.2%**	-	-

† Indicates that CPLEX is unable to solve this instance.

2.5.3 Analysis and interpretation of solutions

Table 2.3 presents magnitude of cost components obtained for instances 9-16 presented in Table 2.2. These cost components, listed in column 1, are: total cost incurred, cost per metric ton of switchgrass delivered ($/Mg) (15% moisture content), and cost per dry Mg of switchgrass produced ($/ dry Mg). This column also presents number of load-out equipment sets, and tractor-trailers required for each of the instances. Thus, for the original data set with 48 weeks, 199 SSLs, a plant requirement of 6050 Mg per week, and a telehandler load-out rate of 605 Mg per week, the total cost incurred is $3,937,595, with 10 load-out equipments sets and 24 tractor-trailers.

Table 2.3: Cost description and analysis for instances 9-16

Cost Components	**Ins. 9**	**Ins. 10**	**Ins. 11**	**Ins. 12**	**Ins. 13**	**Ins. 14**	**Ins. 15**	**Ins. 16**
1. Total cost incurred ($)	$2,649,245	$3,103,206	$3,612,972	$3,610,853	$3,470,173	$3,937,595	$4,382,855	$4,422,982
2. Cost per Mg of switchgrass delivered ($/Mg)	$14	$14	$15	$14	$13	$14	$14	$14
3. Cost per dry Mg of switchgrass delivered ($/ dry Mg)	$16	$17	$18	$17	$16	$16	$17	$16
4. Number of load-out equipment sets	7	8	9	9	9	10	11	11
5. Number of tractor-trailers	17	19	22	22	21	24	29	29

Figure 2.5 illustrates the routes for the 10 equipment sets purchased for the original data set with 199 SSLs, a plant requirement of 6050 Mg per week, and a telehandler load-out rate of 605 Mg per week. Triangles represent SSLs, and arcs indicate the routes of the load-out equipments over each week. The number in the arcs indicates the number of equipment sets traversing the arc, and a number inside each triangle indicates the SSL number. Numbers located over these triangles and inside () indicate the amount of biomass shipped to the bio-energy plant in a given week, while the numbers located under these triangles and inside () represent the number of tractor-trailers allocated to the corresponding SSL. For instance, two equipment sets (E1 and E2) start at the plant at time 0, and move to SSL 9. Then, in

week 1, E1 and E2 load-out 1210 Mg of biomass into the racks that are then hauled by 6 tractor-trailers. These tractor-trailers perform back-and forth trips between SSL 9 and the bio-energy plant. Then, at the end of week 1, E1 and E2 are moved to SSL 6. Thus, in week 2, both E1 and E2 unload 1210 Mg of biomass into racks that are hauled by 5 tractor trailers to the bio-energy plant. At the end of week 2, again both E1 and E2 together are moved to SSL 10. In week 3, 1210 Mg of biomass is loaded into racks that are hauled by 6 tractor-trailers. At the end of week 3, E1 is moved to SSL 3, while E2 is moved to SSL 54. In week 4, E1 loads-out 605 Mg of biomass from SSL 3 that are hauled by 2 tractor-trailers. An additional equipment set, namely E3, is moved to SSL 54 as well, and both E2 and E3 load-out a total of 1210 Mg hauled by 5 tractor-trailers. The figure, thus, displays the routes for each equipment over 48 weeks.

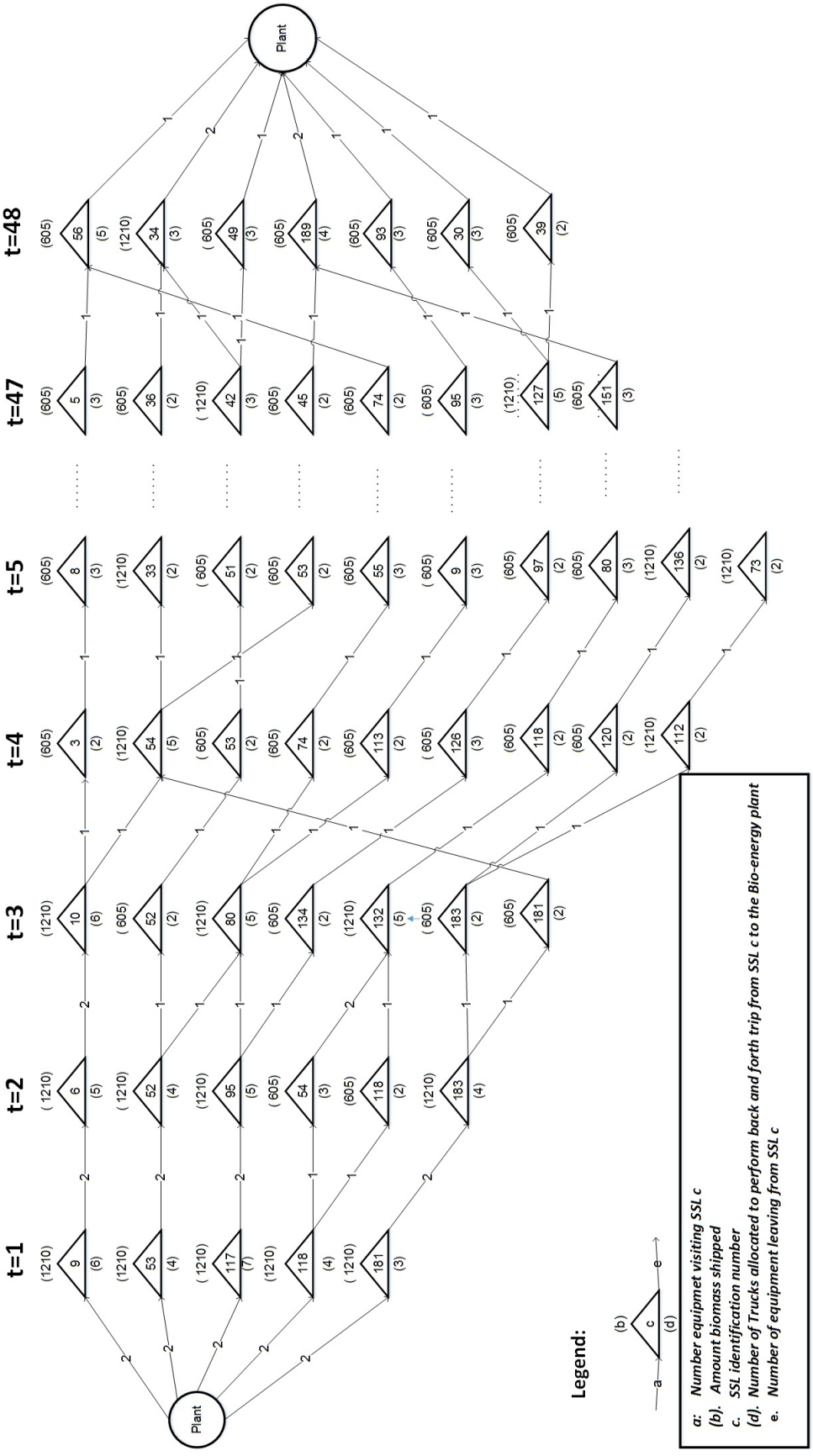

Figure 2.5: 10 equipment routes for the instances with 199 SSLs and plant requirement of 6050 Mg per week (Instance 14).

Chapter 3

Corn Stover Harvesting Scheduling Problem (CSHSP)

3.1 Introduction

Cellulosic ethanol, produced from agricultural residues (e.g, straw or corn stover) or energy crops (e.g, miscanthus or switchgrass), is expected to replace a percentage of petroleum fuel used in the transportation sector, and, at the same time, reduce greenhouse gas emissions in the Unites States. The Energy Independence and Security Act (EISA) of 2007 has established a minimum volume of renewable fuels required for blending with transportation fuel, which increases from 9 billion gallons in 2008 to 36 billions gallons by 2022. By that year, 44.4% of the 36 billion gallons of renewable fuel must be produced from cellulosic ethanol in the United States in order to achieve the desired reduction in greenhouse gas emissions from the transportation sector.

Corn stover, a residue of the corn grain harvest, has been identified as a feedstock for the production of cellulosic ethanol (Perlack et al. (2005)). Corn is the largest U.S crop in terms of total production, and corn stover is the biomass used for two cellulosic ethanol plants

that are in the process og being commissioned. These plants are the Poet plant located in Emmetsburg, Iowa (POET-DSM (2014)), expected to ultimately use 258,500 dry Mg/year, and the DuPont plant located in Nevada, Iowa (DuPont (2015)), that is expected to use 340,000 dry Mg/year. It is very important to develop effective management tools to perform the corn stover harvesting season, and thus, make cellulosic ethanol economically competitive with fossil fuels.

The corn stover harvesting takes place only after the grain harvesting has been completed. Once the grain harvesting has been completed, the farmers want the stover to be removed as quickly as possible so that they can proceed with the (fall) operations needed to prepare the fields for spring planting. Both the POET and DuPont facilities employ several companies to perform the corn stover harvesting, and these companies consist of several harvest crews. (These crews will hereafter be referred to as "custom" harvesters.) Each crew has the set of equipment needed to collect the stover and transport it to a storage location, typically known in the Midwest supply systems as "roadside storage". At a later time, this stover is loaded onto trucks for transport to the plant. This system eliminates the need for corn producers to own and maintain expensive harvesting equipment that they will use only for a short 5-week harvest season. The centralized management of the harvest requires the plant to coordinate and assign the custom harvesters to fields as they are "called in" at the plant. This study focuses on this problem, and, in particular, on developing cost-effective schedules for the custom harvesters that can be used in practice by the plant.

Two different harvesting technologies are used. In one technology, the stover is harvested in rectangular bales ($1\ m \times 1.22\ m \times 2.44\ m$), and in the other stover is harvested in round bales (1.83 m diamter $\times$ 1.52 m long). Capacities of the balers are different and this is considered in the analysis. Furthermore, multiple balers of each technology are available, and a given field can be visited by multiple balers of the same technology at the same time. Currently, round and rectangular balers cannot operate in a given field at the same time.

This chapter is organized as follows: In Section 3.2, we present a description of the harvesting system under study and a brief literature review of related problems. In Section 3.3, we propose a mixed integer programming formulation for the static CSHSP, and in Section 3.5, we present an algorithm to solve the dynamic CSHSP. Computational results for application of this methodology to a real-life dataset are presented in Section 3.6

3.2 System description, problem statement, and a brief literature review

3.2.1 System description

Typically, corn grain is harvested and the residue, corn stover, is left behind on the fields. Once all the grain has been extracted, the farmers call the biofuel plant to schedule harvest of the corn stover. The plant contracts with custom operators to harvest the stover. These operators own the equipment to harvest these fields, which typically include equipment for shredding, windowing, baling, and in-field hauling. The infield-hauling equipment moves the bales from where they are dropped in the field to a roadside storage location. The cost associated with growing the crop and harvesting the grain is the responsibility of the feedstock producer. They are paid for the stover lying in the field. The only cost we include in this study is the cost to collect the stover and place the bales in roadside storage.

The corn stover harvest season is divided into T time periods. The model presented in the sequel is generic and able to handle different time periods (minutes, hours, days, or weeks), but, in this study, and for the sake of clarity, we have defined the length of a period as one day. A workday is assumed to have 8 productive hours. A productive hour is defined as the time when the baler is actually baling stover; that is, it does not include maintanence time or operator breaks. Thus, the harvest season corresponds to a given number of productive

days.

Several requirements must be considered while harvesting a given field. (1). Once a field becomes available, it should be harvested within a given number of periods (due date). If a field is not harvested before its due date, we assume a penalty cost of $100 per period past its due date. (2). All fields must be completely harvested over the harvest season. (3). Balers cannot leave a field until the harvesting has been completed. Current rules for commercial operation do not allow a baler to leave a field and return later to complete the harvest.

Recapping earlier discussion, there are two baling technologies used. The rectangular bale technology produces a bale weighting 0.63 Mg, and, typically, the average productivity of this baler is 27 bales per hour. Thus, the capacity of a baler is then 27 bale/h$\times$ 8 h/day $\times$ 0.63 Mg/bale=136 Mg/day. The round bale technology generates round bales weighting 0.60 Mg, and the average productivity is 16 bales per hour. Thus, the capacity of this baler is 16 bale/h $\times$ 8 h/day $\times$ 0.60 Mg/bale=76.8 Mg/day. A day will, hereafter, be referred to as a "period".

At the beginning of the harvest season, all balers are located at a central depot, and at the end of each time period, balers can stay at their current locations, come back to the depot, or be transported directly to a new production field. In this study, we incorporate a baler routing cost, which is defined as the cost to transport the baler to the next field to be harvested. We assume a fixed cost of $200 per move and a variable cost of $2.29 per kilometer of travel between fields.

Total routing cost for a given baler (rectangular or round) is the sum of the cost for all moves during the season. Total routing cost paid by the plant is the sum of the total routing cost for all the balers.

3.2.2 Problem statement, and brief literature review

The static corn stover harvesting problem (CSHSP) can be concisely stated as follows: Given a set of F pre-located productions fields, a_f Mg of corn stover available in field $f \in F$, a time window, $w_f = [r_f, d_f]$, where r_f and d_f are the ready time and due date for field f, B types of balers with capacity k_b (Mg per period), and a planning horizon of length T, determine the optimal number of balers, the type of bales to produce in field $f \in F$, the required number of periods to complete harvesting of field f that depends on the harvesting technology assigned to this field, and the routing of balers over the planning horizon. The goal is to minimize the total cost incurred by the plant, which consists of the following: (a) capital cost incurred by purchasing balers, (b) contract price ($/Mg) paid to custom harvesters by the plant, (c) due-date penalties paid to the feedstock producers for late harvesting, and (d) the total routing cost incurred by the balers.

The definition of the dynamic CSHSP is identical to that of the static CSHSP, except that the fields become available dynamically over the planning horizon (fields are called in each day over the harvesting season). It is assumed that the number of balers is known *a priori*. This number is determined at the beginning of the analysis by solving the static CSHSP using the dataset for all the fields under contract by the plant.

The CSHSP can be represented as a time-dependant heterogeneous multiple high-multiplicity traveling salesmen problem with known time windows for harvesting each field (TDHHMmT-SPTW). The time-dependent feature emerges because the harvest season is assumed to be split into T time periods. The heterogeneous characteristic emerges due to different baling technologies (round or rectangular), and the multiple salesmen corresponds to the use of several balers. High-multiplicity occurs because a field might require several visits to complete its harvesting. The number of periods to complete the harvest of a given field depends on the capacity of the baling technology chosen, and the number of balers assigned to at field. The time window is the number of periods between the ready time and the due date for each

field. To the best of our knowledge, this is a new problem, and it has not been studied in the literature. However, several individual features have been addressed.

The time-dependent traveling salesman problems (TD-TSP), where the cost associated with an arc depends of its position in the tour with respect to the base city, have been addressed by Picard and Queyranne (1978), Fox et al. (1980) and Gouveia and Voß (1995).

The traveling salesman problem with time windows (TSPTW) is an extension of the classical TSP, where each city must be visited within a time window defined by the earliest and latest arrival times. The TSPTW has been studied by Dumas et al. (1995), Pesant et al. (1998), Ascheuer et al. (2000), and Calvo (2000), among others. The concept of time window in the CSHSP is different from that in the TSPTW since in the former a time window enforces the periods in which a city (field) can be harvested instead of their arrival times that depend on the traveling distance between the cities.

The high multiplicity traveling salesman problem (HMATSP) is a generalization of the TSP wherein cities must be visited multiple times. It has been studied by Grigoriev and van de Klundert (2006) and Sarin et al. (2011). They, respectively, propose an exponential-length representation and a tight polynomial-length formulation with flow-based representation for subtour elimination constraints. Nevertheless, the CTHSP is different from the HMATSP by the fact that, in the former, the number of visits to each field or city (n_f) is determined endogenously by the model, while in the HMATSP, n_f is a given input parameter.

3.3 Mathematical formulation of the static CSHSP

In this section, we formulate the static CSHSP as a mixed-integer programming model. Consider the following notation.

Set and indices:

F : Set of production fields.

F^0 : Set of production fields including the base denoted by 0.

B : Set of balers (round or rectangular).

T : Set of periods.

Parameters:

a_f : Total corn stover available in field f, $\forall f \in F$ (Mg).

k_b : Harvesting capacity of baler b. $b \in B$ (Mg/period).

r_f : Ready time of field f, $f \in F$.

d_f : Due date of field f, $f \in F$.

c_b^{hi} : Capital cost incurred by purchasing baler $b \in B$ (\$/unit).

c_{tbij}^{mo} : Mobilization cost incurred to transport baler b from field i to field j in period t, $i, j \in F^0$, $t \in T$ (\$).

c_b^{ha} : Cost incurred for baler type b for harvesting in each period (\$/period), $b \in B$.

c^{la} : Cost incurred for harvesting a field after its due date (\$/period).

m_f : Maximum number of balers that can visit field f in a given period.

Decision variables:

w_{bf} : Binary variable that equals to 1 if field f produces bales type b; and 0 otherwise, $\forall b \in B$, $\forall f \in F$.

y_{tf} : Binary variable that equals to 1 if field f is harvested in period t; and 0 otherwise, $\forall f \in F$, $\forall t \in T$.

x_{tbij} : Number of balers type b traversing arc (i, j) at the end of period t, $i, j \in F^0$, $t \in T$, $b \in B$.

z_{tbf} : Number of balers type b allocated to field f during period t, $\forall f \in F$, $\forall b \in B$, $\forall t \in T$.

n_b : Total number of balers type b required over the planning horizon (harvest season), $b \in B$.

s_f : The harvest starting period for field f, $\forall f \in F$.

e_f : The harvesting ending period for field f, $\forall f \in F$.

l_f : The number of periods elapsed if field f is harvested after its due date (used to calculate the late-harvesting penalty), $\forall f \in F$.

e_{max} : The actual length of the harvest season.

The following mixed-integer programming formulation is proposed to capture the static CSHSP (S-CSHSP):

Model S-CSHSP:

$$\text{Minimize} = \sum_{b\in B} c_b^{hi} n_b + \sum_{t\in T}\sum_{b\in B}\sum_{f\in F} c_b^{ha} z_{tbf} + \sum_{f\in F} c^{la} l_f + \sum_{t\in T}\sum_{b\in B}\sum_{i\in F}\sum_{j\in F} c_{tbij}^{mo} x_{tbij} \quad (3.1)$$

subject to

$$\sum_{b\in B} w_{bf} = 1, \quad \forall f \in F \quad (3.2)$$

$$\sum_{t\in T} z_{tbf} = \lceil \frac{a_f}{k_b} w_{bf} \rceil, \quad \forall b \in B, \quad \forall f \in F \quad (3.3)$$

$$\sum_{b\in B} z_{tbf} \leq m_f y_{tf}, \quad \forall t \in T, \quad \forall f \in F \quad (3.4)$$

$$n_b \geq \sum_{f\in F} z_{tbf}, \quad \forall t \in T, \quad \forall b \in B \quad (3.5)$$

$$y_{tf} = 0, \quad f \in F, \quad \forall t \in T : t < r_f \quad (3.6)$$

$$y_{tf} \leq \sum_{b\in B} z_{tbf}, \quad \forall t \in T, \quad \forall f \in F \quad (3.7)$$

$$s_f \leq t + T(1 - y_{tf}), \ \forall t \in T, \ \forall f \in F \quad (3.8)$$

$$e_f \geq t y_{tf}, \quad \forall t \in T, \quad \forall f \in F \quad (3.9)$$

$$l_f \geq e_f - d_f, \quad \forall f \in F \quad (3.10)$$

$$s_f \leq e_f, \quad \forall f \in F \quad (3.11)$$

$$e_{max} \geq e_f, \quad \forall f \in F \quad (3.12)$$

$$\sum_{t\in T} y_{tf} = e_f - s_f + 1, \quad \forall f \in F \quad (3.13)$$

$$\sum_{j\in F} x_{tbij} = z_{tbi}, \quad \forall t \in T, \quad \forall b \in B, \quad \forall i \in F \quad (3.14)$$

$$\sum_{f\in F} x_{0b0f} = n_b, \quad \forall b \in B \quad (3.15)$$

$$\sum_{f\in F} x_{Tbf0} = n_b, \quad \forall b \in B \quad (3.16)$$

$$\sum_{j \in F} x_{(t-1)bjf} - \sum_{j \in F} x_{tbfj} = 0, \quad \forall t \in T : t \geq 1, \quad \forall b \in B, \quad \forall f \in F \tag{3.17}$$

$$w_{bf} \in \{0,1\}, \quad \forall b \in B, \quad \forall f \in F \tag{3.18}$$

$$y_{tf} \in \{0,1\}, \quad \forall t \in T, \quad \forall f \in F \tag{3.19}$$

$$n_b \geq 0 \text{ and integer}, \quad \forall b \in B \tag{3.20}$$

$$z_{tbf} \geq 0, \quad \forall t \in T, \quad \forall b \in B, \quad \forall f \in F \tag{3.21}$$

$$s_f, \ e_f, \ l_f \geq 0, \quad \forall f \in F \tag{3.22}$$

$$e_{max} \geq 0. \tag{3.23}$$

$$x_{tbfi} \in z^+, \quad \forall t \in T, \quad \forall b \in B, \quad \forall f \in F, \quad \forall i \in F \tag{3.24}$$

The objective function, Equation (3.1), minimizes sum of the capital cost incurred by purchasing baler, the operation cost of balers, the penalty cost for late harvest, and the mobilization cost. Constraints (3.2) enforce that each field produces a unique type of bale, and Constraints (3.3) enforce that if a baler type b visits field f, then the field is fully harvested by requiring the baler to visit f exactly $\lceil a_f/K_b \rceil$ periods (total Mg of field f, a_f, divided by the baler capacity per period, k_b). Constraints (3.4) enforce that field f can be visited by at most m_f balers at the same time if field f is harvested in period t. Constrains (3.5) capture the number of baler type b required for the harvest season. Constraints (3.6) ensure that field f is not harvested before its ready time (r_f). Constraints (3.7) enforce that harvesting can occur at field f in period t only if there is a baler available. Constraints (3.8) and (3.9) capture, respectively, the actual harvesting starting and ending periods for field f. Constraints (3.10) capture the tardiness incurred in the harvesting of field f, and Constraints (3.11) enforce a logical implication to assure that the harvest starting period for field f is less than its ending period. Constraints (3.12) capture the actual length of the planning horizon (length of harvest season). Constrains (3.13) impose that a baler can move to another field different than f only when f is fully harvested, a condition used in practice. Note that these constraints enforce that a field f is harvested during its starting and ending period, and thus eliminating the possibility of balers to move from f before it is completely harvested. Constraints (3.14) enforce that field i in period t is visited by z_{tbi} baler type b. Constraints (3.15) and (3.16) make sure n_b balers depart and return to the base at the start and end of the harvest season.

Constrains (3.17) are the standard flow conservation constraints, and they model the mobilization feature of the problem though T periods. Finally, Constraints (3.18)-(3.23) capture the domains of the decision variables.

3.4 A solution approach for the static CSHSP

Our preliminary computational results revealed that the model S-CSHSP is a very difficult to solve by CPLEX as it was unable to generate a feasible solution to most of the instances considered within two hours of wall clock time. We, therefore, exploit the structure of the problem and design a method to obtain an advanced-start solution to help reduce the computational effort taken by CPLEX, which is shown to be very effective. We call this approach CPLEX with advance start (CPLEX+AS). To this end, note that Constraints (3.2)-(3.13) and Constrains (3.15)-(3.17) model, respectively, the harvest and routing decisions. Constraints (3.14) are the linking constraints combining the two components. CPLEX+AS exploits this structure as follows:

1. *Step1:* Obtain a solution to a modified mathematical model given by the objective function (3.25) and Constraints (3.2)- (3.13), and (3.18)-(3.23). Note that this model only considers harvesting decisions and overlooks completely the routing feature.

$$\text{Minimize} = \sum_{b\in B} c_b^{hi} n_b + \sum_{t\in T}\sum_{b\in B}\sum_{f\in F} c_b^{ha} z_{tbf} + \sum_{f\in F} c^{la} l_f \tag{3.25}$$

2. *Step2:* Generate routing decisions to the solution obtained in Step 1 by solving the S-CSHSP model by considering harvesting decisions to be fixed to these values obtained in Step 1. Thus, this approach obtains a feasible solution to the S-CSHSP.

3. *Step3:* Solve model S-CSHSP with the advanced-start solution obtained in Step 2 until the optimal solution is obtained or a prescribed time limit is reached.

3.5 A solution approach for the dynamic CSHSP

In this section, we describe in detail the proposed algorithm to solve the CSHSP dynamically in a rolling horizon fashion.

. Consider the following additional notation:

Additional notation:

B'	:	Set of available balers for round or square balers.
b'	:	A unique baler index, $b' \in B'$.
T_{max}	:	Maximum temporary length of the planning horizon (harvest season).
t	:	Current time period.
L_t	:	List of active fields ready to be harvested in period t.
$s_{b't}$	:	Initial field assignment of baler b' in period t.
$l_{b't}$	:	Location of baler b' in period t.
$k_{b'}$	:	Capacity of baler b'.
a_{tf}	:	The Mg available for harvesting in production field f in period t.

The proposed dynamic approach solves model S-CSHSP at execution of the method described in Section 3.4 each increment of time that new information become available for a given horizon of lenght T'. The method is described in Figure 3.1. In Step 1, the termination condition is checked. Steps 2-22 are executed only if fields are available at period t. In Steps 2 and 3, all fields available in period t are added to the list of active field (L_t). In Steps 4-8, the nopreemption feature of the problem is imposed by assigning balers to the fields which have not been fully harvested. In Step 5, for each equipment $b' \in B'$, if $s_{b't} \neq \emptyset$, then b' is assigned to $s_{b't}$, i.e, b' has been assigned to field s'_b in previous periods, but b' has not harvested field s'_b completely yet. In Step 9, a new schedule is generated by solving model S-CSSHP (presented in Section 3.3), where current location of equipment and noopreemption of productions fields are incorporated accordingly. In Steps 10-20, the feedback of the schedule generated and executed is received, and parameters are updated. In Steps 10-13, for each baler $b' \in B'$, we check if it is assigned to field f. In Step 13, if the capacity of the baler ($k_{b'}$) is less than the Mg available in field f in period t (a_{tf}), then in Step 14, the

total biomass available is decreased by $k_{b'}$, the baler location ($l_{b't}$) is set to f, and the equipment is assigned to f in the next period ($s_{b't} = f$) so that b' harvests field f completely before moving to a different field. Otherwise, we proceed to Step 16, where field f is dropped from L_t since it is fully harvested. Finally, in Steps 21-22, respectively, t is increased by one and statistics related to the operational and mobilization cost are updated.

On the other hand, if new fields are not available in period t, then Steps 23-44 are executed. In Step 24, if L_t is not empty, then in Step 25 the schedule generated in the previous period is executed. Steps 26-36 are identical to Steps 10-20. Otherwise, we perform Steps 38-42, where we check, for each baler, its location ($l_{b't}$). If any $l_{b't} \neq 0$, then we assign it to depot. Finally, in Steps 43-44, respectively, t is increased by one and the statistics are updated.

In the dynamic algorithm, at every time period when a new schedule is generated, we specify, as an input parameter, a temporary length of the planing horizon, T', to solve model S-CSHSP formulation, which can be computed as follows:

$$T' = \lceil \frac{\sum_{f \in L} a_{tf}}{\sum_{b' \in B'} k_{b'}} \rceil \tag{3.26}$$

In Equation (3.26), the numerator and denominator represent, respectively, the total biomass available (Mg) and the total capacity of balers in period t (Mg/period). Thus, this value indicates the number of periods needed to harvest all the fields, which is then rounded-up to the nearest integer since T' is discrete.

3.6 Computational results

In this section, we investigate performances of the methods proposed for the solution of the static CSHSP and dynamic CSHSP presented in Sections 3.3 and Section 3.5, respectively. These methods were coded using Microsoft Visual Studio Professional 2012 with CPLEX as solver. All runs were made on a 3.1 GHz computer having 32 GB of RAM.

```
1: while t < T_max do
2:   if a field f ∈ F arrives in period t then
3:     Add f to L
4:     for b' = 1 to |B'| do
5:       if s_{b't} ≠ ∅ then
6:         Assign baler b' to field s_{b't} in period t
7:       end if
8:     end for
9:     Generate a new schedule by solving CSHSP model
10:    for b' = 1 to |B'| do
11:      for f = 1 to |L| do
12:        if Equipment b' is assigned to field f then
13:          if field k_{b'} < a_{tf} then
14:            Set a_{t+1f} = a_{tf} − k_{b'}, s_{b't} = f, and l_{b't} = f.
15:          else
16:            a_{t+1f} = 0, s_{b't} = ∅, l_{b't} = f, and Drop f from L
17:          end if
18:        end if
19:      end for
20:    end for
21:    Let t = t + 1
22:    Update statistics
23:  else
24:    if L ≠ ∅ then
25:      Follows schedule generated in period t − 1
26:      for b' = 1 to |B'| do
27:        for f = 1 to |L| do
28:          if Equipment b' is assigned to field f then
29:            if field k_{b'} < a_{tf} then
30:              Set a_{t+1f} = a_{tb} − k_{b'}, s_{b't} = f, and l_{b't} = f.
31:            else
32:              a_{t+1f} = 0, s_{b't} = 0, l_{b't} = ∅, and Drop f from L
33:            end if
34:          end if
35:        end for
36:      end for
37:    else
38:      for b' = 1 to |B'| do
39:        if b' ∈ B' is not located at 0 then
40:          Set l_{b't} = 0
41:        end if
42:      end for
43:      Let t = t + 1
44:      Update statistics
45:    end if
46:  end if
47: end while
```

Figure 3.1: Algorithm to solve the dynamic CSHSP.

3.6.1 Results for the static CSHSP

We present results for the static CSHSP, where fields ready times are assumed to be all known at the beginning of the harvest season. We also present a sensitivity analysis on estimated parameters.

Real-life scenario

The initial dataset utilized to illustrate the use of the proposed methodology consists of a real-world data set provided by Antares Group Inc (consulting engineering firm with a speciality in renewable energy projects). It consists of data from the 2014 corn stover harvest season collected by the company. Generally described, it consists of 85 fields, and the following information for each field: the location (latitude and longitude), the area of field (hectares), the ready times (r_f), and the types of bales that can be produced in each field (round or rectangular). Each field due date was 7 periods from their ready time (r_f). Figures 3.2 and 3.3 display the field locations and the number of fields becoming available per day, respectively, in the dynamic environment.

Other parameters used in the model are as follows: round and rectangular balers capital costs were estimated to be \$100,000 and \$140,000, respectively. Contract prices paid to round and rectangular balers were estimated to be 21.60 \$/dry Mg and 23.60 \$/dry Mg, which are reported in Sokhansanj and Turhollow (2002).

We solved this real-life scenario described above using CPLEX and CPLEX with the advanced-start solution (CPLEX+AS) described in Section 3.4 with a time limit of two hours of wall clock time. CPLEX obtained a feasible solution with an objective value of \$ 81,309,608 (Z_{IP}) with an integer optimality gap (gap) (defined as $100 * |(Z_{IP} - Z_{LB})/Z_{IP}|\%$, where Z_{LB} is the best lower bound obtained during the optimization process) of 99.53%. On the other hand, CPLEX+AS obtained a solution with an objective value of \$669,950 and a gap of 0.05%, and thus, clearly outperforming CPLEX.

The schedule generated by CPLEX+AS for the 2014 corn stover harvest season is displayed in Figure 3.4. We present, for each baler, the field to be harvested during each day and the routes to follow over the harvest season. Note that a baler travels from field i to j ($i \neq j$) at the end of

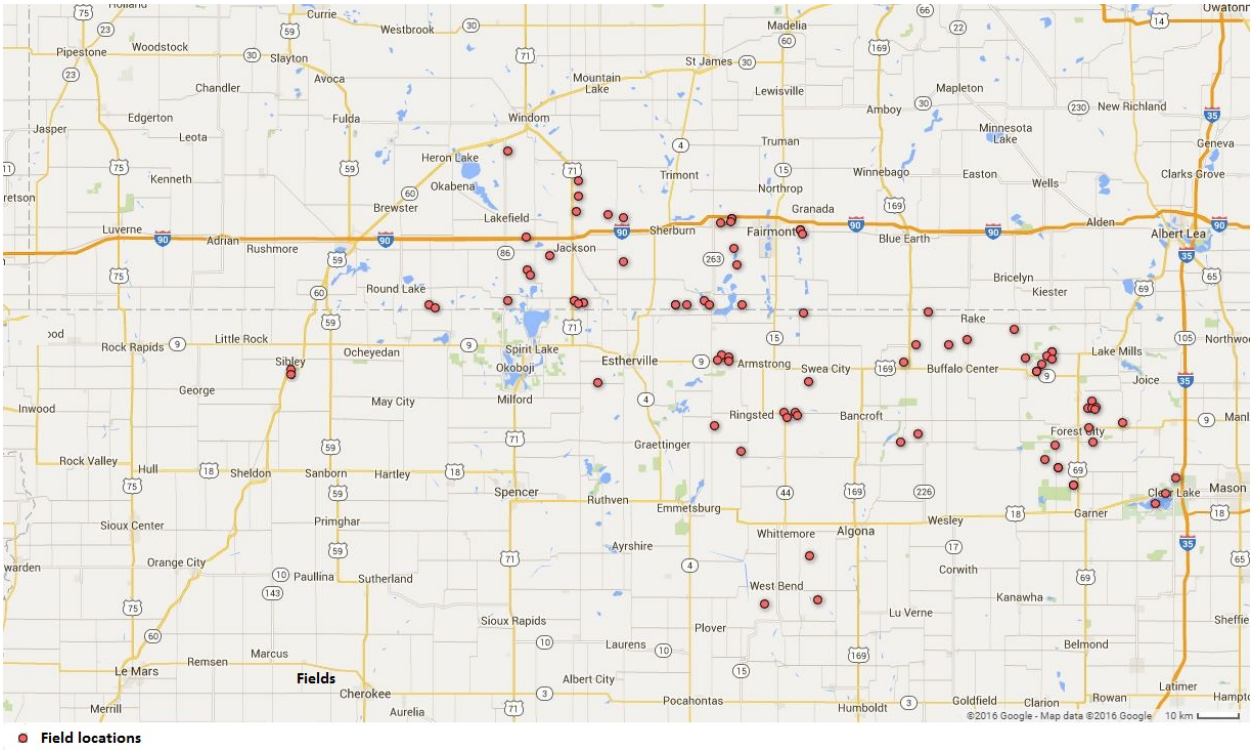

Figure 3.2: Geographical distribution of the corn stover production fields harvested during the 2014 season for the given data. (Source: Google Maps. Only used for reference purpose.)

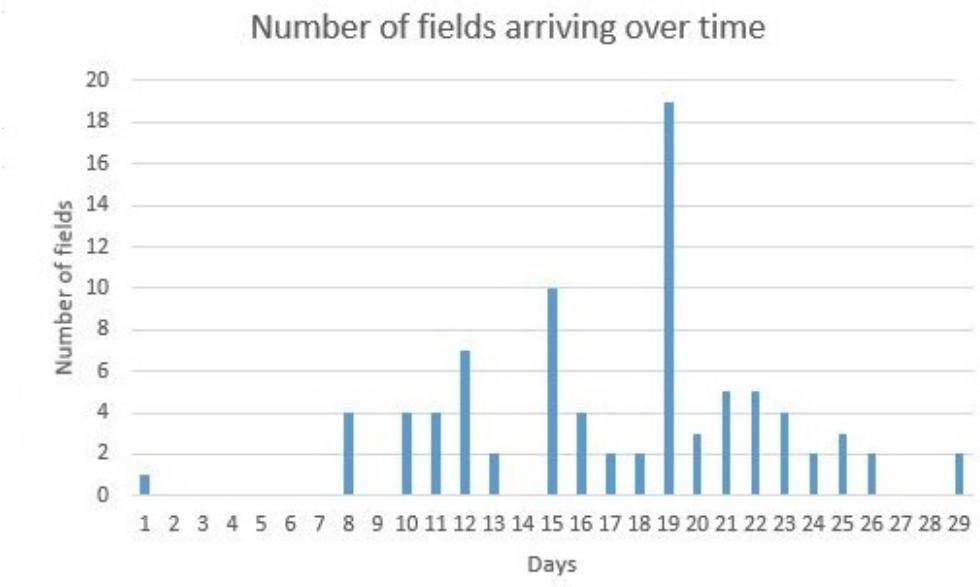

Figure 3.3: Number of fields becoming available over time for the given data collected during the 2014 season.

a day, and if $i = j$, then the baler stays at location i. This scenario considered requires 2 round and 3 rectangular balers. The round balers, represented by B^{ro}, start their routes by harvesting Fields 10 and 9 on days 11 and 10, and finish their assignments by visiting Fields 73 and 74 on day 30. On the other hand, the rectangular balers, represented by B^{re}, start their routes by harvesting Fields 3, 5, and 1 on days 9, 8, and 2, and finish by harvesting Fields 84, 82, and 83 on days 36, 33, and 33, respectively. This solution incurs a total cost of \$669,950, where the balers' capital cost, the balers' contract price paid for harvesting, the balers' routing cost, and the cost incurred for tardiness harvesting represent 92.5%, 5.5%, 1.9%, and 0.0% of the total cost, respectively. Finally, note that all fields are harvested before their due date, and thus, the tardiness cost is \$0 (we impose a penalty of \$ 100 per day if a field is harvested after its due date).

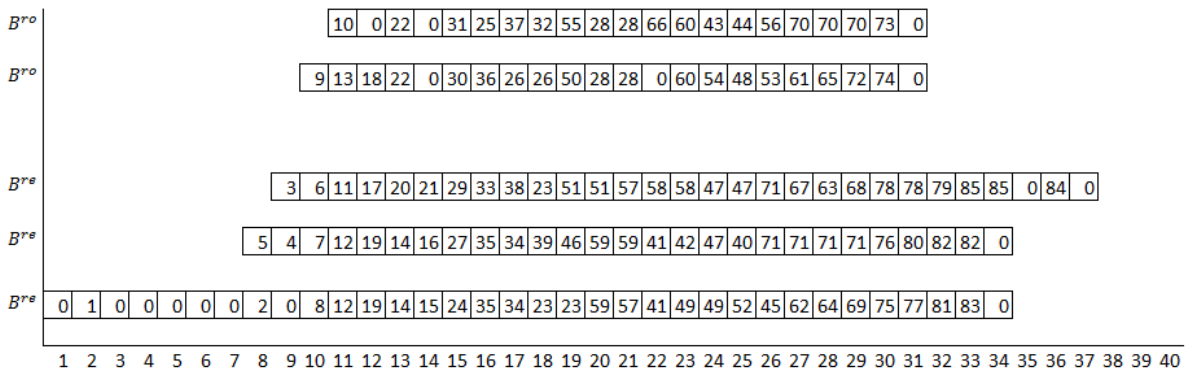

Figure 3.4: Schedule generated for each baler for the data given in the 2014 season.

In the static CSHSP model, parameters values are assumed to be all known with certainty. However, in practice, due to uncertainty, the parameter are difficult to estimate accurately such as field ready times and the capacity of the custom harvesters. Next, we study the impact of these parameters values in the problem.

Sensitivity analysis: effects of changing the field ready times

Due to weather related events, it might be difficult to forecast field ready times, and thus, we perform a sensitivity analysis on field ready times to determine its impact on the complexity of the problem and the solutions obtained. To this end, we derive two sets of additional instances from

the original data set by modifying the field ready times: Set I, denoted by FD1-FD4, is generated by creating field ready times (r_f) from a fitted distribution using the original data set; while Set II, denoted by UD5-UD8, is created by sampling field ready times from a uniform distribution $U[R_{min}, R_{max}]$, where R_{min} and R_{max} are the smallest and largest field ready times in the original data set.

Table 3.1 presents the impact on problem complexity when field ready times are changed. Column 1 presents the instance, and Columns 2-4 indicate the best lower bound (Z_{LB}), the best integer solution(Z_{IP}), and the total CPU time in seconds or the optimality gap (gap) obtained by CPLEX. Columns 5-7 are identical to Columns 2-4, and they present the results obtained by CPLEX+AS. The symbols "-" and "tl" denote that a feasible solution could not be obtained for that instance in the prescribed time limit. If only a feasible solution was obtained within the time limit, then we report the gap value. CPLEX was unable to obtain a feasible solution for any of the instances, while CPLEX+AS obtained solutions to all the instances with a maximum gap of 0.47%. From this results, we can infer that CPLEX+AS outperformed CPLEX, and that from the view point of problem complexity, changing field ready times did not impact the performance of CPLEX+AS. From now on, we only consider CPLEX+AS in our investigation.

Table 3.2 shows the impact on the solutions when field ready times are modified. Column 1 indicates the instance, Columns 2-3 denote, respectively, the number of round and rectangular balers required over the harvest season, and Columns 4-8 presents cost description for each problem, where each column represents the capital cost, the operation cost, the routing cost, the tardiness for late harvesting, and the total cost, respectively. We observe that the total number of balers required during the harvest season for the instances derived from a fitted distribution (FD1-FD4) varies from 5 to 6, and, on the instances derived from a uniform distribution (UD1-UD4), this number is 4. From the cost perspective, we observe that the average total cost is \$625,723, and the average mobilization cost is \$11,898. Note that the capital cost represent on average 92% of the total cost. From these results, we can assert that a change in field ready times impacts the number of balers required for the harvest season.

Sensitivity analysis: effects of changing the capacity of the custom harvesters

Table 3.1: Impact on the complexity of the static CSHSP when field ready times are change.

	CPLEX			*CPLEX+AS*		
Instance	Z_{LB}	Z_{IP}	T/gap	Z_{LB}	Z_{IP}	T/gap
FD1	375099	-	tl	766517	767862	0.18%
FD2	404246	-	tl	726735	728347	0.22%
FD3	406301	-	tl	711392	712106	0.10%
FD4	416164	-	tl	727031	727807	0.11%
UD1	348749	-	tl	484452	486716	0.47%
UD2	388098	-	tl	525901	527209	0.25%
UD3	335528	-	tl	525268	527746	0.47%
UD4	346218	-	tl	525743	527994	0.43%

Table 3.2: Impact on the solutions of the static CSHSP when the field ready times are varied.

			Cost description			
Instance	n_{round}[†]	$n_{\text{rectangular}}$[‡]	*Capital*	*Contract price*	*Mobilization*	*Tardiness*
FD1	3	3	$ 720,000	$ 36,800	$ 11,062	$ 0
FD2	4	2	$ 680,000	$ 35,400	$ 12,947	$ 0
FD3	1	4	$ 660,000	$41.000	$ 11,106	$ 0
FD4	4	2	$ 680,000	$ 35,200	$ 12,607	$ 0
UD1	3	1	$ 440,000	$ 35,000	$ 11,716	$ 0
UD2	2	2	$ 480,000	$ 34,000	$ 13,209	$ 0
UD3	2	2	$ 480,000	$ 36,800	$ 10,946	$ 0
UD4	2	2	$ 480,000	$ 36,400	$ 11,594	$ 0

[†] Number of round balers; [‡] Number of rectangular balers

Table 3.3: Impact on the complexity of the static CSHSP when the capacity of custom harvesters is changed.

Instance	k_{round} †	$k_{\text{rectangular}}$ ‡	Z_{LB}	Z_{IP}	T/gap
O	76	136	669618.39	669950	0.05%
	72	129	712323.68	712894	0.08%
	68	122	770237.74	771936	0.22%
FD1	76	136	766517.26	767862	0.18%
	72	129	770331.64	770717	0.05%
	68	122	814177.68	814748	0.07%
FD2	76	136	728128.5	728347	0.03%
	72	129	767901.01	769517	0.21%
	68	122	770218.01	773156	0.38%
UD1	76	136	484452.08	486716	0.47%
	72	129	526205.59	528903	0.51%
	68	122	528562.27	531432	0.54%

† Capacity of round balers; ‡ Capacity of rectangular balers

We also vary the capacity of custom harvesters to study its effect in the complexity of the problem and solutions. For the baseline scenario, we have considered the capacities of the rectangular and round balers to be 136 Mg/day and 76 Mg/day, respectively. We varied this capacity to be 95% and 90% of the expected value. Table 3.3 shows the impact on the problem complexity when these capacities are changed, where Columns 2-3 denote the round and rectangular balers' capacity used, respectively, and the remaining columns are identical to those presented in the previous table. Note that the instance denoted by O represents the instance with the original ready times. We can see that CPLEX+AS obtained solutions within 0.54% optimality gap. Therefore, changing the harvester capacities did not impact the performance of CPLEX+AS.

Table 3.4 illustrates the impact on the solution when the capacity of the custom harvesters is varied. We observe that as the capacity of the custom harvesters decrease, both the number of balers and the total cost increase. As example, consider instance O, where in the baseline scenario, we assume the capacities of the round and rectangular balers to be 76 and 136 Mg/day, respectively. Given this scenario, the total number of required balers is 5, but if we decrease these two capacities by 10%, then the total number of required balers increases by 1 and the total cost increases by $ 101,986.

Table 3.4: Impact on the solutions of the static CSHSP when the capacity of the custom harvesters is varied.

					Cost description			
Instance	k_{round}	$k_{\text{rectangular}}$	n_{round}	$n_{\text{rectangular}}$	*Capital*	*Contract price*	*Mobilization*	*Tardiness*
0	76	136	2	3	$ 620,000	$ 37,000	$ 12,950	$ 0
	72	129	1	4	$ 660,000	$ 41,400	$ 11,194	$ 0
	68	122	3	3	$ 720,000	$ 39,600	$ 12,336	$ 0
FD1	76	136	3	3	$ 720,000	$ 36,800	$ 11,062	$ 0
	72	129	3	3	$ 720,000	$ 37,400	$ 13,317	$ 0
	68	122	2	4	$ 760,000	$ 41,600	$ 13,148	$ 0
FD2	76	136	4	2	$ 680,000	$ 35,400	$ 12,947	$ 0
	72	129.2	3	3	$ 720,000	$ 37,800	$ 11,717	$ 0
	68	122.4	3	3	$ 720,000	$ 40,200	$ 12,956	$ 0
UD1	76	136	3	1	$ 440,000	$ 35,000	$ 11,716	$ 0
	72	129.2	2	2	$ 480,000	$ 38,000	$ 10,903	$ 0
	68	122.4	2	2	$ 480,000	$ 40,000	$ 11,432	$ 0

Sensitivity analysis: effect of changing the due dates

We study the impact of changing the field due dates. In the baseline scenario, we assume a due date of 7 days after the ready time for each field. We study the impact of having shorter harvesting time windows by decreasing this parameter to 3 days, and a wider time windows by increasing this parameter to 10 days. Table 3.5 shows the impact on the problem complexity when field due dates are changed. Column 2 indicates the due date parameter used, and the remaining columns are identical to those presented in the previous tables. Note that CPLEX+AS solved all instances within 0.63% optimality gap. Overall, we observe that by having shorter time windows, the problem become easier to solve (those instances can be solve to optimality). We also notice that changing the field due dates did not impact the performance of CPLEX+AS.

Table 3.6 presents the impact on the solutions when field due dates are changed. We remark that as the width of time windows reduces, the model tends to use solutions with larger number of custom harvesters, and therefore, solutions with shorter time windows incur higher cost. To see this, consider instance 0 on this table. When each time window has width of 3 days, 2 round and 5 square balers are required. If we increase the width of each time windows to 10 days, 3 round and 2 square balers are only needed. Furthermore, the difference in the cost incurred between these solutions is $325,242.

Table 3.5: Impact on the complexity of the static CSHSP when the field due dates are changed.

		CPLEX+AS		
Instance	Due date	Z_{LP}	Z_{IP}	*T/gap*
O	3	952388	952388	235.09
	7	669618.39	669950	0.05%
	10	624487.63	627146	0.42%
FD1	3	1009846	1009846	3764.30
	7	766517.26	767862	0.18%
	10	665801.26	668393	0.39%
FD2	3	953179	953179	475.73
	7	726734.83	728347	0.22%
	10	625449.05	629407	0.63%
UD1	3	628612.42	629200	0.09%
	7	484452	486716	0.47%
	10	427819.87	432210	0.56%

Table 3.6: Impact on the solutions of static CSHSP when the field due dates are varied.

				Cost description			
Instance	*Due date*	n_{round}	$n_{\text{rectangular}}$	*Capital*	*Contract price*	*Mobilization*	*Tardiness*
O	3	2	5	$ 900,000	$ 38,000	$ 14,388	$ 0
	7	2	3	$ 620,000	$ 37,000	$ 12,950	$ 0
	10	3	2	$ 580,000	$ 35,400	$ 11,746	$ 0
D1	3	4	4	$ 960,000	$ 36,200	$ 13,746	$ 0
	7	3	3	$ 720,000	$ 36,800	$ 11,062	$ 0
	10	2	3	$ 620,000	$ 37,600	$ 10,793	$ 0
D2	3	2	5	$ 900,000	$ 39,400	$ 13,779	$ 0
	7	4	2	$ 680,000	$ 35,400	$ 12,947	$ 0
	10	3	2	$ 580,000	$ 35,400	$ 14,007	$ 0
R1	3	3	2	$ 580,000	$ 36,200	$ 13,000	$ 0
	7	3	1	$ 440,000	$ 35,000	$ 11,716	$ 0
	10	1	2	$ 380,000	$ 38,200	$ 12,010	$ 0

3.6.2 Results for the dynamic CSHSP

In this section, we present results obtained by solving the dynamic CSHSP, where fields become available throughout the harvest season, and harvesting and routing decisions must be made with partial information. We solve the dynamic CSHSP using the periodic reoptimization scheme al-

gorithm presented in Section 3.5, where the model S-CSHSP is solved at different decisions point throughout the harvest season.

Dynamic or online algorithms are evaluated based on its competitive ratio (see details in Pinedo (2012)), which is defined as a worst-case ratio between of the objective values obtained by the dynamic algorithm and by the offline approach, where all data is known *a priory*. The main drawback on this measure of effectiveness is that requires proof of a bound, which might be difficult to determine in real world problems. To overcome this difficulty, Mitrović-Minić et al. (2004) have introduced the value of the information, V^I, which is based on empirical result and defined by:

$$V^I = \frac{z^{*\text{ online}} - z^{*\text{ offline}}}{z^{*\text{ online}}} \tag{3.27}$$

where $z^{*\text{ online}}$ and $z^{*\text{ offline}}$ denote the optimal objective values of the solutions obtained by the dynamic and offline (static) approach. Since it might be difficult to determine $z^{*\text{ online}}$ and $z*^{*\text{ offline}}$, (3.27) is replaced by:

$$V_a^I = \frac{z_a^{\text{online}} - z_a^{\text{offline}}}{z_a^{\text{online}}} \tag{3.28}$$

where z_a^{online} and z_a^{offline} denote, respectively, the objective values obtained the dynamic and static problem when solved using the same approach, a.

In our application, z_a^{offline} is replaced by the objective value obtained by CPLEX-AS when solving the model S-CSHSP, and z_a^{online} is replaced by the objective value obtained by the dynamic algorithm. Therefore, we can interpret the value of the information as a gap between the solution obtained by the static and dynamic approach.

Table 3.7 presents results obtained on the base case scenario as well as the randomly generated instances generated in the previous section. In this experiment, for each instance, we fix the number of balers to those reported in Table 3.1, where Columns 4-7 present the cost description for the solutions obtained by the dynamic algorithm. Column 8 denotes the value of the information, V_a^I, computed as described in (3.28). We notice that V_a^I varied between 20.1% and 23.0%, and its average value was 21.0%. Therefore, on average, the proposed dynamic approach can generate

solutions with a cost 21.0% higher than the cost obtained by the static approach, where field ready times are assumed to be known forehand.

From a practical point of view, the number of baler is a strategic decision that must be made at the beginning of the harvest season, and it cannot vary as assumed in the previous table. Therefore, we fix the number of balers to those obtained by solving the initial real life case (2 round and 3 rectangular balers), and thus, the dynamic algorithm must react to this initial decision. Table 3.8 presents results obtained for the dynamic case under this scenario, where Column 8 shows increment in total cost when compared with the costs obtained in Table 3.7, where the number of balers was changed for each instance. The remaining columns are identical to those presented in previous tables. We observe that by fixing the number of required balers, the cost of the solution increases mainly due to the cost incurred by mobilization and tardiness for late harvesting. Overall, by keeping fixed the number of balers, the proposed dynamic algorithms, on average, was able to obtain solutions that increase the total cost by 4.42% when compared to those in which the number of balers is varied.

Table 3.7: Results obtained by solving the dynamic CSHSP when the number of balers is fixed to those obtained in Table 3.2.

Instance	n_{round}	$n_{\text{rectangular}}$	*Contract price*	*Mobilization*	*Tardiness*	*Total*	V_a^I
O	2	3	$ 39,400	$ 18,936	$ 4,600	$ 62,936	20.6%
FD1	3	3	$ 38,600	$ 20,748	$ 2,800	$ 62,148	23.0%
FD2	4	2	$ 37,400	$ 19,973	$ 3,700	$ 61,073	20.8%
FD3	1	4	$ 42,200	$ 19,680	$ 3,400	$ 65,280	20.2%
FD4	4	2	$ 36,200	$ 20,930	$ 3,600	$ 60,730	21.3%
UD1	3	1	$ 36,400	$ 17,965	$ 4,100	$ 58,465	20.1%

Table 3.8: Results obtained by solving the dynamic CSHSP when the number of balers is kept fixed to that obtained in the real-life scenario.

			Cost description				
Instance	n_{round}	$n_{\text{rectangular}}$	*Contract price*	*Mobilization*	*Tardiness*	*Total*	% Increase
O	2	3	$ 39,400	$ 18,936	$ 4,600	$ 62,936	0.00%
FD1	2	3	$ 40,000	$ 21,911	$ 11,300	$ 73,211	17.80%
FD2	2	3	$ 38,400	$ 19,551	$ 3,900	$ 61,851	1.27 %
FD3	2	3	$ 38,400	$ 19,885	$ 3,400	$ 61,685	-5.51%
FD4	2	3	$ 38,400	$ 17,637	$ 6,200	$ 62,237	2.48 %
UD1	2	3	$ 40,000	$ 24,382	$ 200	$ 64,582	10.46%

Chapter 4

Solving the Single and Multiple Asymmetric Traveling Salesmen Problems by Generating Subtour Elimination Constraints from Integer Solutions

4.1 Introduction

The traveling salesman problem (TSP) is one of the most well-known combinational optimization problems studied in the literature, and hundred of papers and several books have been devoted to its study. It consists of determining a tour that starts and ends at a given base city after having visited a set of cities exactly once. The TSP can be classified into symmetric TSP (STSP) or asymmetric TSP (ATSP), where in the former and latter it is assumed that the distance or cost to travel between any two cities is symmetric or asymmetric, respectively. Practical applications

of the TSP include the drilling of printed circuit boards, X-ray crystallography, overhauling gas turbine engines, the order-picking problem in warehouses, computer wiring, clustering of a data array, vehicle routing, and production scheduling, among others (Reinelt (2003)).

An extension of the ATSP, which has not received much attention, is the multiple asymmetric traveling salesmen problem (mATSP), where multiple salesmen are located at the depot. Some applications of this problem reported in the literature include print press scheduling, crew scheduling, school bus routing, interview scheduling, hot strip-rolling scheduling, and design of global navigation satellite surveying network (Bektaş (2006)).

In this chapter, we present an algorithmic approach for solving the single and multiple asymmetric traveling salesmen problems (ATSP and mATSP) by generating subtour elimination constraints (SECs) from integer solutions. Typically, the generation of SECs is performed from fractional points by solving a min-cut problem (see Padberg and Rinaldi (1990)). The idea of generating SECs from integer points was first used in Dantzig et al. (1954) to solve the symmetric traveling salesman problem (STSP), where the authors added SECs by visually inspecting a given solution. In Miliotis (1976) and Miliotis (1978), SECs were also generated from integer points, but in contrast with Dantzig et al. (1954), an algorithmic process was proposed to automatically generate SECs. Recently, the STSP has also been solved in Pferschy and Staněk (2016) using this idea with the aim of exploiting the effectiveness of current commercial solvers, where instances involving up to 400 cities were solved with reasonable effort. The similarity between our method and these previous studies is that we also generate SECs from integer points; however, our approach differs in the algorithmic process employed to accomplish this. In Dantzig et al. (1954), Miliotis (1976), Miliotis (1978), and Pferschy and Staněk (2016), whenever new SECs are generated, the current relaxed problem is solved to optimality. On the other hand, in our approach, we generate SECs as lazy-cuts, which are invoked whenever an integer solution to the problem is identified in any part of the branch-and-bound tree during the process of implicitly solving the original problem, where the procedure ultimately terminates with a proven optimal solution. We also differ from Dantzig et al. (1954), Miliotis (1976), Miliotis (1978), and Pferschy and Staněk (2016) in that we solve the ATSP and mATSP instead of the STSP, which are relatively more challenging classes of problems.

The remainder of this chapter is organized as follows. In Sections 4.2 and 4.3, we present exponential and polynomial-length formulations as well as describe the proposed approaches to generate SECs from integer points for the ATSP and mATSP, respectively. In Section 4.4, we present computational results for the proposed approach when applied to large-sized ATSP and mATSP-instances involving between 100 and 1001 cities, and compare its performance against the most effective formulations reported in the literature for both ATSP and mATP when solved directly by CPLEX. We also compare our results with the most effective state-of-the-art exact algorithms reported in Roberti and Toth (2012) for solving the ATSP.

4.2 The asymmetric traveling salesman problem (ATSP)

The ATSP can be concisely defined as follows: Given a complete graph with vertex set N in which city 1 denotes the base city, and an asymmetric distance matrix $[c_{ij}]$, $i, j \in N$, determine an optimal tour that starts and ends at the base city after having visited city i exactly once, $\forall i \in N$, while minimizing the total distance traveled.

4.2.1 Formulations for the ATSP

Let x_{ij} be a binary variable that equals 1 if city i directly precedes city j in the ATSP solution, and zero otherwise, $\forall i, j \in N,\ i \neq j$. Then, the ATSP can be formulated as follows:

$$\textbf{ATSP: Minimize} \sum_{i \in N} \sum_{j \in N,\ j \neq i} c_{ij} x_{ij} \tag{4.1}$$

subject to :

$$\sum_{j \in N,\ j \neq i} x_{ji} = 1, \quad \forall i \in N \tag{4.2}$$

$$\sum_{j \in N,\ j \neq i} x_{ij} = 1, \quad \forall i \in N \tag{4.3}$$

Subtour elimination contraints (SECs) (4.4)

$$x_{ij} \in \{0, 1\}, \quad \forall i, j, \in N,\ i \neq j. \tag{4.5}$$

The objective function (4.1) minimizes the total cost, and Constrains (4.2) and (4.3), respectively, enforce that each city i is visited and departed from once. Constraints (4.4) are subtour elimination constraints (SECs), which we discuss next.

The SECs have typically been formulated as an exponential set of constraints or a polynomial number of restrictions by introducing additional variables. Dantzig et al. (1954) introduced the following exponential number of SECs:

$$\sum_{i \in N'} \sum_{j \in N',\ j \neq i} x_{ij} \leq |N'| - 1, \quad \forall N' \subset N, N' \neq \emptyset. \tag{4.6}$$

By combining (5.2) and (4.6), Constraints (4.6) can be re-written as:

$$\sum_{i \in N'} \sum_{j \in N - N'} x_{ij} \geq 1, \quad \forall N' \subset N, N' \neq \emptyset. \tag{4.7}$$

Reviews of exact methods for solving the ATSP have been presented in Laporte (1992), Fischetti et al. (2002), D'Ambrosio et al. (2010), and Roberti and Toth (2012). Generally, branch-and-bound and branch-and-cut algorithms relax Constraints (4.6), and generate these SECs on the fly as needed along with several additional facet-defining inequalities. A recent survey of the most effective exact approaches for the ATSP is presented in Roberti and Toth (2012), where the branch-and-bound algorithms of Carpaneto et al. (1995) and Fischetti and Toth (1992), and the branch-and-cut methods of Fischetti et al. (2003) and Applegate et al. (2006) (Concorde) are compared on instances involving up to 443 cities. The results exhibit that branch-and-cut algorithms are most effective in solving these problems.

A comparative analysis of polynomial-length SEC-based formulations for the ATSP appears in Öncan et al. (2009) and Roberti and Toth (2012). In particular, Roberti and Toth (2012) empirically compared several compact formulations and concluded that the three most effective formulations from the viewpoint of a direct solution by CPLEX are MTZ (Miller et al. (1960)), GG (Gavish and Graves (1978)), and DL (Desrochers and Laporte (1991)). These are presented next and will be used as a benchmark for comparison with the proposed approach in the sequel.

Letting u_i indicate a real number representing the order in which city i is visited in an optimal

tour, $\forall i \in N - \{1\}$, the MTZ-SECs can be stated as follows:

$$u_i - u_j + (|N| - 1)x_{ij} \leq |N| - 2, \quad i, j \in N - \{1\}, \quad i \neq j \tag{4.8}$$

$$1 \leq u_i \leq |N| - 1, \quad \forall i \in N - \{1\}. \tag{4.9}$$

Letting y_{ij} be a real variable indicating flow in arc (i, j), the GG-SECs are given by:

$$y_{ij} \leq (|N| - 1)x_{ij}, \quad i, j \in N, \quad i \neq j \tag{4.10}$$

$$\sum_{j \in N,\ j \neq i} y_{ji} - \sum_{j \in N',\ j \neq i} y_{ij} = 1, \quad \forall i \in N - \{1\} \tag{4.11}$$

$$y_{ij} \geq 0, \quad i, j \in N, \quad i \neq j. \tag{4.12}$$

The MTZ-SECs were strengthened by Desrochers and Laporte (1991) as follows (denoted by DL-SECs):

$$u_i - u_j + (|N| - 1)x_{ij} + (|N| - 3)x_{ij} \leq |N| - 2, \quad i, j \in N - \{1\}, \quad i \neq j \tag{4.13}$$

$$-u_i + (|N| - 3)x_{i1} + \sum_{j \in N - \{1\},\ j \neq i} x_{ji} \leq -1, \quad i \in N - \{1\} \tag{4.14}$$

$$u_i + (|N| - 3)x_{1i} + \sum_{j \in N - \{1\},\ j \neq i} x_{ij} \leq |N| - 1, \quad i \in N - \{1\}. \tag{4.15}$$

4.2.2 Proposed approaches to generate SECs from integer points for the ATSP

We propose to solve the ATSP by generating SECs from integer points. The proposed approaches (described below) were implemented using CPLEX's lazy constraints callback function, where specific violated SECs (if any) are generated whenever an integer solution to the current relaxed problem is identified (see details in IBM (2016)). These approaches are implemented in a traditional branch-and-cut scheme, where the presence of subtours is verified whenever an integer-feasible solution has been identified. If this solution does not involve subtours, then we obtain an incumbent solution. Otherwise, SECs are added to the model, and the linear programming (LP) relaxation of the problem is solved. After adding SECs, we have three possibilities. (1) We obtain an integer solution containing no subtours, in which case, we have an incumbent solution. (2) We obtain an integer solution containing subtours, in which case we add the SEC and repeat the process, and (3) we obtain a non-integer solution in which case the solver continues solving the LP relaxation of the problem until an integer solution is obtained. Thus, the procedure finally terminates after having

identified an optimal solution to the original problem by enforcing the SECs only when they are needed. In this sense, the SECs are implemented as lazy cuts.

The identification of all subtours for a given assignment solution satisfying (4.2)-(4.3), and (4.5) is straightforward. For any such x-solution, we identify all node subsets N_i, $i = 1, ..., s$, which form disjoint connected subgraphs or subtours, where the set N_1 contains the base city, $\{1\}$, and s represents the total number of such subtour-based node sets in the current solution. We explore several approaches for generating SECs at any step as described below.

Approach 1 (A_1): Append a single cut of type (4.6) based on a set having the largest cardinality.

Approach 2 (A_2): Generate all s cuts of type (4.6).

Approach 3 (A_3): Generate all s cuts of type (4.7).

Approach 4 (A_4): Generate a single cut of type (4.7) based on the set N_1.

Note that Approaches A_2 and A_3 are identical, except for the structure of the (equivalent) cuts generated. Our motivation for testing both these procedures is to investigate if a standard solver performs differently due to the sparsity structure of the type of cuts added at any step.

We also append to all formulations and proposed approaches the two-city SECs given by:

$$x_{ij} + x_{ji} \leq 1, \quad \forall i, j \in N - \{1\}, i \neq j. \tag{4.16}$$

4.3 Multiple asymmetric traveling salesmen problem (mATSP)

The multiple asymmetric traveling salesmen problem (mATSP) can be concisely defined as follows: Given a complete graph with vertex set N in which city 1 denotes the base city, an asymmetric distance matrix $[c_{ij}]$, $i, j \in N$, and m salesmen located at the base city, determine m tours that start and end at the base city after collectively having visited city i exactly once, $\forall i \in N$, while minimizing the total distance traveled.

Next, we present both exponential and polynomial-length formulations for the mATSP as well as our proposed approaches for generating SECs from integer points as lazy cuts.

4.3.1 Formulations for the mATSP

The mATSP can be formulated as follows:

$$\textbf{mATSP: Minimize} \sum_{i \in N} \sum_{j \in N,\ j \neq i} c_{ij} x_{ij} \tag{4.17}$$

subject to :

$$\sum_{j \in N,\ j \neq 1} x_{1j} = m \tag{4.18}$$

$$\sum_{j \in N,\ j \neq 1} x_{j1} = m \tag{4.19}$$

$$\sum_{j \in N,\ j \neq i} x_{ji} = 1, \quad \forall i \in N - \{1\} \tag{4.20}$$

$$\sum_{j \in N,\ j \neq i} x_{ij} = 1, \quad \forall i \in N - \{1\} \tag{4.21}$$

Subtour elimination contraints (SECs) (4.22)

$$x_{ij} \in \{0, 1\}, \quad \forall i, j, \in N,\ i \neq j. \tag{4.23}$$

Constraints (4.18) and (4.19) enforce that m salesmen depart from and return to the base city. Constraints (4.20)-(4.21) assure that each city other than the base is visited and departed from once. Constraints (4.22) are subtour elimination constraints (SECs), which we discuss next.

It can be shown that Constraints (4.6) are not valid for the mATSP, and they need some refinement. Consider an instance having five cities ($n = 5$) and two salesmen ($m = 2$) with a valid solution given by route 1: $x_{12} = 1$, $x_{23} = 1$, $x_{31} = 1$, and route 2: $x_{14} = 1$, $x_{45} = 1$, $x_{51} = 1$. If we let $N' = \{1, 2, 3\}$, then Constraints (4.6) impose $x_{12} + x_{13} + x_{21} + x_{23} + x_{31} + x_{32} \leq 2$, which cuts off this feasible solution. Therefore, (4.6) needs to be imposed just for subsets of cities not involving city 1 as follows:

$$\sum_{i \in N'} \sum_{j \in N',\ j \neq i} x_{ij} \leq |N'| - 1, \quad \forall N' \subset N - \{1\}, N' \neq \emptyset. \tag{4.24}$$

Therefore, the corresponding exponential-length SECs for the mATSP are Constraints (4.7) and (4.24).

A review of the polynomial-length formulations for the mATSP appears in Kara and Bektaş (2006) and Sarin et al. (2014a), where the latter presents a comprehensive review of 32 formulations for the mATSP, and demonstrates that the most effective formulations for the mATSP when solved directly by CPLEX are, again, MTZ, GG, and DL as extended by Kara and Bektaş (2006) to the mATSP, which are presented next.

The MTZ-SECs adapted to the mATSP are as follows:

$$u_i - u_j + (|N| - m)x_{ij} \leq |N| - m - 1, \quad i, j \in N - \{1\}, \quad i \neq j \tag{4.25}$$

$$1 \leq u_i \leq |N| - m, \quad \forall i \in N - \{1\}. \tag{4.26}$$

The GG-SECs adapted to the mATSP are as follows:

$$y_{ij} \leq (|N| - m)x_{ij}, \quad i, j \in N, \quad i \neq j \tag{4.27}$$

$$\sum_{j \in N,\ j \neq i} y_{ji} - \sum_{j \in N',\ j \neq i} y_{ij} = 1, \quad \forall i \in N - \{1\} \tag{4.28}$$

$$y_{ij} \geq 0, \quad i, j \in N, \quad i \neq j. \tag{4.29}$$

The DL-SECs adapted to the mATSP by Kara and Bektaş (2006) are given by (denoted by KB):

$$u_i + (m - |N| - 2)x_{1i} - x_{i1} \leq |N| - m - 1, \quad i \in N - \{1\} \tag{4.30}$$

$$u_i + x_{1i} + x_{i1} \geq 2, \ i \in N - \{1\} \tag{4.31}$$

$$u_i + u_j + (|N| - m)x_{ij} + (|N| - m - 2)x_{ji} \leq |N| - m - 1, \ i, j \in N - \{1\}, i \neq j. \tag{4.32}$$

4.3.2 Proposed approaches to generate SECs from integer points for the mATSP

Similar to the ATSP, we propose to solve the mATSP by generating SECs from integer points. The identification of all subtours for a given set of x_{ij}-values satisfying (4.18)-(4.21) and (4.23) is straightforward. For any such x-solution, we identify all node subsets N_i, $i = 1, ..., s$, which form disjoint connected subgraphs, with N_1 containing $\{1\}$ and the others representing subtours, where s represents the total number of such node sets in the current solution. We test several approaches for generating SECs at each step as described below.

Approach 5 (A_5): Append a single cut of type (4.24) based on the set having the largest cardinality

from $N_2, ..., N_s$.

Approach 6 (A_6): Generate $s-1$ cuts of type (4.24) based on the sets $N_2, ..., N_s$.

Approach 7 (A_7): Generate s cuts of type (4.7) based on the sets $N_1, ..., N_s$.

Approach 8 (A_8): Generate a single cut of type (4.7) based on the set N_1.

We also add to all formulations and proposed approaches the two-city SECs (4.16).

4.4 Computational Results

The proposed algorithms were coded using Microsoft Visual Studio Professional 2012, with CPLEX 12.5.1 as the solver with default parameters. The direct formulations were solved using OPL with default parameters. All runs were made on an Intel Xeon E5-2687W 3.1GHz computer having 1 GB of RAM and running Windows 7, with a maximum permissible CPU time limit of 3,600 seconds.

Table 4.1 displays the 37 test instances used in our experimentation. The columns indicate the instance name (name), the number of cities (n), the source from where the instance was taken (source), and the optimal cost reported in the literature (optimal ATSP cost), respectively.

4.4.1 ATSP

In Table 4.2, we compare the performances of the proposed approaches (A_1, A_2, A_3, and A_4) and the direct solution by CPLEX of the polynomial-length formulations for the ATSP (DL, GG, and MTZ). For each approach, the respective columns indicate the integer solution obtained (Z_{IP}) and the CPU time in seconds to reach optimality or the integer optimality gap after 3600 seconds (T/gap). The symbols "-" and "tl" denote that a feasible solution could not be obtained for that instance in the prescribed time limit (tl=3600 seconds). If a feasible solution was obtained within the time limit, then we report the gap. Approaches A_2 and A_3 outperformed the other methods by solving 30 out of the 33 ATSP instances to optimality, while the DL, GG, and MTZ formulations were able to solve to optimality only 12, 16, and 13 out of the 33 instances, respectively. For the

three instances (atex8, dc895, and dc932) that neither A_2 nor A_3 were able to solve to optimality, the minimum and maximum gaps obtained for the former and latter were 0.05% and 2.52%, and 0.12% and 2.87%, respectively. The DL, GG, and MTZ formulations were unable to obtain a feasible solution to 8, 11, and 7 out of 33 cases, respectively, or to generate feasible solutions for instances exceeding 563 cities; only DL obtained a feasible solution to the 600-city problem (atex8), but it yielded an optimality gap of 20.14% at termination. Finally, we have highlighted the minimum (T/gap) in bold for each instance.

As for comparing the results of the most effective proposed approach, A_2, with the state-of-the-art exact algorithms presented in Roberti and Toth (2012), we acknowledge the difficulty in making a fair comparison due to the following. Roberti and Toth (2012) reported the results obtained by Fischetti et al. (2002) who used a Digital Alpha 553 MHz computer with CPLEX 6.5.3 as the LP solver, whereas we have used a 3.1 GHz computer having 1 GB of RAM along with Microsoft Visual Studio Professional 2012 with CPLEX (12.5.1) as the solver (using default parameters) with a fixed time limit of 3,600 seconds. Nevertheless, Table 4.3 presents a comparison of CPU times consumed by the algorithms presented in Roberti and Toth (2012), namely, the branch-and-bound methods reported in Carpaneto et al. (1995) (CDT) and Fischetti and Toth (1992) (FT), and the branch-and-cut algorithms presented in Fischetti et al. (2003) (FLT), and Applegate et al. (2006) (Concorde). We only show results for the 17 instances involving over 100 cities from Roberti and Toth (2012), and note that the authors only solved instances involving up to 443 cities. The proposed approach outperformed the four exact algorithms in terms of the CPU time required in 10 out of the 17 instances (the minimum time for each instance is highlighted). We point out that Concorde and FLT were able to solve all instances to optimality in Roberti and Toth (2012), but their time limit was set to 10,000 seconds. The time limit for CDT and FT was set to 1,000 seconds, and this might be why CDT and FLT have the maximum number of instances reaching the time limit (tl) in Table 4.3. However, A_2 obtained optimal solutions for all instances within 121.2 seconds. Note that the CPU times reported in columns labelled CDT, FT, FLT, and Concorde were obtained using a slower machine; accordingly, following Roberti and Toth (2012), if we multiply the CPU times reported for A_2 by 10, then the average times for A_2, FLT, and Concorde are 212.9, 223.0, and 1,126.6 seconds. From these results, we can infer that the proposed approach is competitive in

solving real-world ATSP instances. Besides, it does not require the use of a specialized algorithm and can be easily implemented using an off-the-shelf software such as CPLEX.

4.4.2 mATSP

Instances for the mATSP were derived from those presented in Table 4.1. For the sake of brevity, we have selected a subset of these problem instances with the number of salesmen ranging from 2 to 4. In Table 4.4, we compare the performances of the proposed approaches (A_5, A_6, A_7, and A_8) with the direct solution by CPLEX of the most effective compact formulations for the mATSP (KB, GG, and MTZ). The column designations in Table 4.4 are identical to those in Table 4.2. We have highlighted the minimum (T/gap) valued in bold for each instance. Approach A_6 outperformed the other methods by solving 27 out of the 36 mATSP instances to optimality, while KB, GG, and MTZ were able to solve to optimality only 7, 7, and 8 out of the 36 instances, respectively. Note that A_7 solved to optimality only 26 out of the 36 mATSP instances, but it achieved minimum T/gap in the largest number of instances (22 out of the 36 problems). For those nine and ten instances derived from balas200, atex8, dc985, and dc932 that A_6 and A_7 were unable to solve to optimality, the minimum and maximum gaps for the former and the latter were 0.08% and 4.13% and 0.02% and 2.18%, respectively. We point out that KB, GG, and MTZ were unable to solve instances of size greater than 443 cities except for one instance of atex8 solved by KB to a gap of 17.71%.

We also study the impact of increasing the number of salesmen on the proposed approach A_6 (see Table 4.5), which solved the largest number of instances to optimality in Table 4.4. The number of salesmen, m, in this experiment is given by $m = \lceil 0.05n + 5i \rceil$, for $i = 0, ..., 3$ for each n. The approach A_6 was able to solve 16 out of the 20 instances to optimality, and on those four instances derived from atex8 for which A_6 failed to obtain optimal solutions, the maximum optimality gap was 2.64%. Note that instances derived from atex8 could not be solved to optimality in Table 4.4 as well. Overall, note that increasing the number of salesmen did not impact the performances of A_6.

4.4.3 Discussion

Given the results presented above for the ATSP, we can infer the following:

1. The most effective ATSP formulations reported in the literature (DL, GG, and MTZ) could solve problem instances to optimality involving up to 443 cities within one hour of CPU time when optimized directly by CPLEX (12.5.1).

2. The most effective of the proposed approaches were A_2 and A_3, which were able to solve 30 out of the 33 problem instances to optimality. We notice that generating several cuts to break all subtours at once using either Constraints (4.6) or (4.7) is more efficient than only generating a single cut each time SECs are invoked. Table 4.6 presents the number of SECs generated for each approach for some instances presented in Table 4.1. For example, in the problem instance dc176, A_2 and A_3 generated 124 and 253 SECs, respectively, while A_1 and A_4 added $1,720$ and $2,672$ SECs, respectively. This pattern can be observed for the other instances presented in this table as well.

3. The effectiveness of the proposed approach stems from the fact that it efficiently generates a judicious set of tight cuts (SECs), which, as well known, happen to be facet-defining for the ATSP.

In the same vein, the following comments apply for the mATSP:

1. The most effective mATSP formulations reported in the literature (DL, GG, and BK) could solve instances to optimality involving up to 443 cities within one hour of CPU time when solved directly by CPLEX (12.5.1).

2. The most effective of the proposed approaches were A_6 and A_7. Similar to the ATSP, generating several cuts to break all subtours at once using Constraints (4.7) or (4.24) turned out to be more effective than appending only a single cut each time SECs are generated as in approaches A_5 or A_8. Furthermore, increasing the number of salesmen did not impact the performances of A_6.

3. The effectiveness of the proposed approaches A_6 and A_7 is further highlighted by the fact that all mATSP instances of size greater than 443 cities were solved with a maximum optimality gap of only 2.18%.

Table 4.1: Well-known 33 ATSP instances taken from the literature.

Name	*n*	*Source*	*Optimal ATSP Cost*
kro124	100	TSPLIB (1997)	36230
ftv100	101	TSPLIB (1997)	1788
td100.1	101	Cirasella et al. (2001)	268636
Balas108	108	Roberti and Toth (2012)	152
ftv110	111	TSPLIB (1997)	1958
dc112	112	Cirasella et al. (2001)	11109
Balas120	120	Roberti and Toth (2012)	286
ftv120	121	TSPLIB (1997)	2166
dc126	126	Cirasella et al. (2001)	123235
ftv130	131	TSPLIB (1997)	2307
dc134	134	Cirasella et al. (2001)	5612
ftv140	141	TSPLIB (1997)	2420
ftv150	151	TSPLIB (1997)	2611
Balas160	160	Roberti and Toth (2012)	397
ftv160	161	TSPLIB (1997)	2683
ftv170	171	TSPLIB (1997)	2755
dc176	176	Cirasella et al. (2001)	8587
dc188	188	Cirasella et al. (2001)	10225
code198	198	Cirasella et al. (2001)	4541
Balas200	200	Roberti and Toth (2012)	403
code253	253	Cirasella et al. (2001)	106957
td316.10	317	Cirasella et al. (2001)	691502
rbg323	323	TSPLIB (1997)	1326
rbg358	358	TSPLIB (1997)	1163
rbg403	403	TSPLIB (1997)	2465
rbg443	443	TSPLIB (1997)	2720
dc563	563	Cirasella et al. (2001)	25951
atex8	600	Cirasella et al. (2001)	39982
big702	702	Cirasella et al. (2001)	79081
dc849	849	Cirasella et al. (2001)	37476
dc895	895	Cirasella et al. (2001)	107699
dc932	932	Cirasella et al. (2001)	479900
td1000.20	1001	Cirasella et al. (2001)	1242183

Table 4.2: Comparison of the results obtained by the proposed approaches A_1, A_2, A_3, and A_4 with those obtained by the direct solution of the polynomial-length formulations DL, GG, and MTZ for the ATSP by CPLEX

	A_1		A_2		A_3		A_4		*DL*		*GG*		*MTZ*	
Name	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*
kro124	36230	186.95	36230	**1.08**	36230	1.26	36230	109.89	36230	146.16	36230	82.13	36230	71.26
ftv100	1788	4.9	1788	**0.69**	1788	0.87	1788	4.13	1788	22.64	1788	81.37	1788	15.97
td100.1	286636	0.67	268636	**0.11**	268636	0.34	268636	1.15	268636	0.28%	286636	3.17	286636	0.28%
Balas108	177	17.37%	152	**19.33**	152	42.78	152	1108.1	155	15.48%	152	868.66	159	17.53%
ftv110	1958	4.21	1958	1.15	1958	**0.95**	1958	4.02	1958	22.17	1958	136.88	1958	192.13
dc112	11109	21.57	11109	**1.09**	11109	2.01	11109	27.22	1115	1.63%	11109	2573.27	11136	1.87%
Balas120	323	22.45%	286	**76.82**	286	351.8	293	5.43%	297	8.86%	294	5.76%	304	10.86%
ftv120	2166	7.44	2166	1.47	2166	**1.36**	2166	4.26	2166	61.15	2166	129.15	2166	330.13
dc126	123235	1856.29	123235	**3.78**	123235	4.71	123235	490.56	123442	0.92%	123235	389.92	123771	1.14%
ftv130	2307	3.53	2307	1.68	2307	**1.22**	2307	5.71	2307	25.71	2307	221.46	2307	44.59
dc134	5612	55.04	5612	**2.53**	5612	6.47	5612	43.98	5615	1.09%	5612	265.79	5623	1.23
ftv140	2420	10.47	2420	1.26	2420	**0.87**	2420	11.31	2420	26.13	2420	208.28	2420	60.43
ftv150	2611	5.49	2611	1.05	2611	**0.94**	2611	2.12	2611	27.77	2611	348.29	2611	16.1
Balas160	463	23.61%	397	**121.17**	397	325.1	488	28.05%	417	9.81%	411	5.35%	441	12.85%
ftv160	2683	38.98	2683	**2.29**	2683	3.4	2683	17.32	2683	41.89	2683	253.88	2683	332.34
ftv170	2755	61.56	2755	5.94	2755	**4.9**	2755	27.16	2755	39.86	2755	451.25	2755	179.46
dc176	8587	1941.5	8587	**10.31**	8587	19.52	8587	832.23	8603	1.63%	8597	0.15%	8689	2.58%
dc188	10225	769.99	10225	**7.36**	10225	25.15	10225	692.07	10250	2.55%	10227	0.04%	10346	3.36%
code198	4541	8.03	4541	**0.37**	4541	18.38	4541	17.35	4541	223.30	4541	353.34	4541	69.08
Balas200	489	27.71%	403	**74.65**	403	**1267.62**	527	tl	426	12.64%	495	19.91%	579	34.65%
code253	106957	407.12	106957	6.07	106957	**4.41**	106957	510.84	106957	314.32	106957	3.27%	106957	417.21
td316.10	691502	6.69	691502	**6.10**	691502	9.75	691502.00	10.31	691687	0.05%	691502	414.65	711611	2.85%
rbg323	1326	14.43	1326	9.61	1326	**1.06**	1326	19.70	-	tl	-	tl	1359	3.53%
rbg358	1163	18.06	1163	11.34	1163	**2.71**	1163	9.41	1223	5.31%	-	tl	1165	0.60%
rbg403	2465	16.88	2465	**11.43**	2465	62.03	2465	147.25	2465	2499.37	-	tl	2465	1566.95
rbg443	2720	22.82	2720	**21.0**	2720	25.99	2720	23.56	-	tl	-	tl	2797	2.79%
dc563	-	tl	25951	**337.93**	25951	433.50	-	tl	-	tl	-	tl	-	tl
atex8	-	tl	40561	**2.52%**	40709	2.87%	-	tl	47383	20.14%	-	tl	-	tl
big702	79081	395.46	79081	44.40	79081	**39.55**	79081.00	186.55	-	tl	-	tl	-	tl
dc849	37848	0.05%	37476	**224.95**	37476	347.63	37476	1854.12	-	tl	-	tl	-	tl
dc895	-	tl	107740	**0.07%**	107786	0.12%	-	tl	-	tl	-	tl	-	tl
dc932	-	tl	480019	**0.05%**	480019	0.15%	-	tl	-	tl	-	tl	-	tl
td1000.20	1242183	21.28	1242183	**24.74**	124183	60.23	124183	426.58	-	tl	-	tl	-	tl

Table 4.3: Comparison of the CPU times required by the proposed approach A_2 with those reported for the state-of-the-art exact algorithms in Roberti and Toth (2012) for solving the ATSP.

Name	A_2	*CDT*	*FT*	*FLT*	*Concorde*
Kro124	1.1	tl	135.7	**1.0**	9.9
balas108	**19.3**	tl	tl	89.0	1416.0
ftv100	**0.7**	16.6	29.6	2.2	12.6
ftv110	**1.2**	5.0	38.4	7.4	25.6
balas120	**76.8**	tl	tl	1276.3	7186.9
ftv120	**1.5**	50.1	99.4	13.1	54.4
ftv130	1.7	6.4	16.6	**1.6**	16.6
ftv140	**1.3**	13.9	38.8	2.1	25.6
ftv150	**1.1**	3.1	17.0	2.6	27.0
balas160	**121.2**	tl	tl	671.1	7848.0
ftv160	**2.29**	93.8	316.2	3.8	55.7
ftv170	5.9	tl	tl	**4.1**	41.9
balas200	**74.7**	tl	tl	1712.8	2294.2
rgb323	9.6	**0.1**	0.3	0.4	23.9
rgb358	11.3	**0.1**	0.5	0.5	29.3
rgb403	11.4	**0.1**	1.0	1.3	49.3
rgb443	21.0	**0.1**	1.2	1.4	34.5
# Ins. [§]	10	4	0	3	0

[§] Number of instances requiring minimum CPU times.

Table 4.4: Comparison of the proposed approaches A_5, A_6, A_7, and A_8 with the direct solution of the polynomial-length formulations KB, GG, and MTZ for the mATSP by CPLEX.

			A_5		A_6		A_7		A_8		KB		GG		MTZ	
Name	n	m	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*	Z_{IP}	*T/gap*
kro124	100	2	36847	5.09	36847	**0.89**	36847	2.89	36847	123.01	36847	14.74	36847	60.03	368477	49.78
		3	37537	3.96	37537	**1.87**	37537	2.92	37537	60.36	37537	85.41	37537	68.22	37537	33.24
		4	38241	8.07	38241	1.68	38241	**1.15**	38241	31.84	38241	85.97	38241	71.87	38241	29.30
Balas108	108	2	152	51.36	152	**16.85**	152	16.40	152	268.06	154	13.58%	153	5.27%	159	16.29%
		3	155	46.10	155	**8.39**	155	11.36	155	163.18	157	13.28%	155	2.02%	157	9.35%
		4	159	22.26	159	**12.28**	159	12.70	159	288.70	162	13.50%	165	10.23%	162	13.53%
Balas120	120	2	287	3035.12	287	**452.23**	287	1382.90	303	11.97%	293	7.27%	296	4.87%	298	8.77%
		3	290	987.83	290	566.80	290	**493.90**	290	1.10%	290	3.80%	293	4.36%	296	5.97%
		4	294	307.84	294	**182.91**	294	194.95	294	315.790	298	5.24%	294	2027.37	296	3.51%
Balas200	200	2	406	2.35%	403	**2959.10**	403	3019.52	528	30.98%	432	13.20%	459	14.21%	520	27.77%
		3	405	1.13%	405	**1471.70**	405	1507.97	503	27.04%	437	14.28%	521	23.53%	485	22.17%
		4	410	1.04%	410	**2210.37**	410	0.29%	552	33.31%	433	11.79%	491	18.17%	652	40.99%
rbg323	323	2	1341	12.60	1341	6.46	1341	**3.63**	1341	8.63	1341	1.12%	1341	2314.26	1341	1845.12
		3	1356	10.72	1356	6.80	1356	**1.26**	1356	7.50	1356	308.41	1356	1986.89	1356	79.75
		4	1372	12.71	1372	6.43	1372	**4.91**	1372	10.61	1372	139.06	1372	1415.10	1372	171.45
rbg443	443	2	2730	29.40	2730	55.91	2730	7.91	2730	**7.47**	2730	0.37%	-	tl	2730	0.37%
		3	2740	107.20	2740	**9.38**	2740	25.30	2740	39.66	2740	1653.66	-	tl	2740	694.20
		4	2750	7.33	2750	9.03	2750	**2.38**	2750	29.20	2750	954.74	-	tl	2750	1146.19
dc563	563	2	26093	0.28%	26043	279.57	26043	**275.65**	26121	0.39%	-	tl	-	tl	-	tl
		3	26149	0.12%	26135	668.68	26135	**228.45**	-	tl	-	tl	-	tl	-	tl
		4	26259	0.19%	26227	256.23	26227	**156.09**	26240	0.12%	-	tl	-	tl	-	tl
atex8	600	2	43408	12.80%	41279	4.13%	40440	**2.18%**	-	tl	-	tl	-	tl	-	tl
		3	42285	9.78%	41051	2.95%	40547	**1.74%**	-	tl	-	tl	-	tl	-	tl
		4	43356	11.17%	41031	2.05%	40914	**1.88%**	-	tl	46361	17.71%	-	tl	-	tl
big702	702	2	79281	174.67	79281	75.02	79281	**33.48**	79281	2517.36	-	tl	-	tl	-	tl
		3	79841	345.12	79481	233.52	79481	**54.01**	79513	0.05%	-	tl	-	tl	-	tl
		4	79881	319.15	79881	29.71	79881	**26.91**	79881	510.37	-	tl	-	tl	-	tl
dc895	895	2	-	tl	108178	0.16%	108099	**0.05%**	-	tl	-	tl	-	tl	-	tl
		3	-	tl	108510	0.10%	108483	**0.02%**	-	tl	-	tl	-	tl	-	tl
		4	-	tl	108938	0.20%	108896	**0.04%**	-	tl	-	tl	-	tl	-	tl
dc932	932	2	-	tl	480148	0.14%	479960	**0.03%**	-	tl	-	tl	-	tl	-	tl
		3	-	tl	480112	0.13%	480027	**0.04%**	-	tl	-	tl	-	tl	-	tl
		4	-	tl	480218	0.08%	480018	**0.02%**	-	tl	-	tl	-	tl	-	tl
td1000.20	1001	2	1242733	**22.33**	1242733	24.26	1242733	93.46	1242733	28.36	-	tl	-	tl	-	tl
		3	1243403	**47.86**	1243403	54.09	1243403	71.51	1243403	63.20	-	tl	-	tl	-	tl
		4	1244361	104.49	1244361	68.58	1244361	**56.18**	1244361	135.97	-	tl	-	tl	-	tl

Table 4.5: Performance of the proposed approach A_6 on larger mATSP instances with a varying number of salesmen.

Name	*n*	*m*	*T/gap*
Balas108	108	6	4.21
		11	5.32
		16	3.15
		21	1.17
Balas120	120	6	42.68
		11	37.35
		16	8.07
		21	0.01
Balas200	200	10	2849.35
		15	598.33
		20	273.76
		25	550.62
dc563	563	29	114.44
		34	166.02
		39	428.35
		44	136.38
atex8	600	30	1.43%
		35	2.64%
		40	2.00%
		45	1.93%

Table 4.6: Number of SECs generated by each of the proposed approach A_1, A_2, A_3, and A_4 when solving the ATSP.

Name	A_1	A_2	A_3	A_4
kro124	971	46	51	1289
Balas108	4446	194	152	4198
Balas120	3889	222	327	3370
dc126	1721	96	276	1782
Balas160	2866	236	183	3955
dc176	1720	124	253	2672
dc188	766	163	359	1474
Balas200	3483	386	302	4257
code253	325	28	41	604
dc563	669	451	546	1151
atex8	756	456	438	913
big702	79	35	47	71
dc849	234	61	121	235

Chapter 5

High-Multiplicity Asymmetric Traveling Salesman Problem (HMATSP) and its Extensions

5.1 Introduction

The *High-Multiplicity Asymmetric Traveling Salesman Problem* (HMATSP) is a generalization of the traveling salesman problem, where a salesman departs and then returns to the base city after having visited each city multiple times. It can be concisely defined as follows: Given a complete graph with vertex set N, an asymmetric distance matrix $[c_{ij}]$, and positive integers n_i, $i, j \in N$, determine an optimal tour that starts and ends at a base city after having visited city i exactly n_i times, $\forall i \in N$, while minimizing the total distance traveled.

The HMATSP arises during the batch production of a family of products that incur sequence-dependent setup times between the processing of batches of different product types on one or more machines. Other examples include the scheduling of landing airplanes Cosmadakis and Papadimitriou (1984), the CHES problem Baker and Muckstadt (1989), and the primary pharmaceutical

manufacturing scheduling problem Sarin et al. (2014b).

Two existing formulations for the HMATSP have been proposed by Grigoriev and van de Klundert (2006) and Sarin et al. (2011). The former presents a formulation for the HMATSP having an exponential number of subtour elimination constraints, while the latter proposes a multi-commodity polynomial-length formulation with flow-based subtour elimination constraints. In this chapter, we present a single-commodity polynomial-length formulation with flow-based subtour elimination constraints for the HMATSP that, although not as tight as that proposed in Sarin et al. (2011), turns out to be more effective to use amongst the existing formulations for a direct solution by CPLEX. We also use this new formulation to model variants or extensions of the HMATSP that involve multiple traveling salesmen. We call this class of problems as the *High-Multiplicity Multiple Asymmetric Traveling Salesmen Problem* (HMmATSP). We consider different versions of this class of problems pertaining to single and multiple bases. In the single base case, all m salesmen are located at one central base, whereas for multiple bases, m_b salesmen are located at base b. We study the non-fixed destination, multiple base HMmATSP, which requires only a designated number of salesmen, but not particular salesmen, to return to each base. Applications of the HMmATSP include loading-equipment fleet management in biomass logistics. Finally, we present accelerated Benders algorithms that exploit structure of the primal subproblem and are able to solve large-sized instances of the HMATSP and its extensions.

The remainder of this chapter is organized as follows. In Section 5.2, we present the existing model formulations for the HMATSP due to Grigoriev and van de Klundert (2006) and Sarin et al. (2011), and we propose a single-commodity flow-based formulation for the HMATSP and establish its relationship to the latter formulation. We also develop Benders-based algorithms for the formulations presented in Sarin et al. (2011) and in this chapter and use an acceleration technique to improve their convergence. In Section 5.3, we transform the HMATSP to an equivalent ATSP representation with the aim of using effective formulations developed in the literature for the ATSP. In Section 5.4, we demonstrate the versatility of our new formulation by modeling several extensions of the HMATSP, and propose exact solution approaches to solve these problems to optimality. In Section 5.5, we present computational results for the proposed solution approaches.

5.2 Formulations for the HMATSP

The existing two formulations for the HMATSP in the literature are an exponential-length formulation presented in Grigoriev and van de Klundert (2006) and a polynomial-length formulation proposed in Sarin et al. (2011). In this section, we present both of these formulations, and then, propose an alternative model formulation for the HMATSP along with comparisons and motivation.

Consider the following notation:

Parameters:

- N : Set of cities, where node 1 is the base city.
- n_i : Number of visits required for city i, $\forall i \in N$, with n_1=1.
- c_{ij} : Distance from city i to j, $\forall i, j \in N$.

Decision Variables:

- x_{ij} : Number of times city i directly precedes city j in the HMATSP solution, $\forall i, j \in N$.
- p_{ij}^u : Binary variable that equals 1 if commodity u, which is required to traverse from node 1 to node u, $\forall u \in N - \{1\}$, flows along the way from node $i \in N$ to $j \in N - \{1\}$, $j \neq i$; and equals 0, otherwise.

5.2.1 Existing Formulations

We refer to the formulation presented in Grigoriev and van de Klundert (2006) as HMATSP-GK, which is stated as follows:

$$\textbf{HMATSP-GK:}\ \text{Minimize} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \tag{5.1}$$

subject to :

$$\sum_{j \in N} x_{ji} = n_i, \quad \forall i \in N \tag{5.2}$$

$$\sum_{j \in N} x_{ij} = n_i, \quad \forall i \in N \tag{5.3}$$

$$\sum_{i \in N'} \sum_{j \in N - N'} x_{ij} \geq 1, \quad \forall N' \subset N, N' \neq \emptyset \tag{5.4}$$

$$x_{ij} \geq 0 \text{ and integer}, \quad \forall i, j \in N. \tag{5.5}$$

The objective function (5.1) minimizes the total distance traveled. Constraints (5.2) and (5.3) enforce that each city i is visited (and exited) exactly n_i times. Note that self-loops are permitted, thus, we do not impose $i \neq j$. Constraints (5.4) are the subtour elimination constraints that prohibit the occurrence of disconnected circuits, and Constraints (5.5) represent logical requirements.

It has been shown in Sarin et al. (2011) that Constraints (5.4) are equivalent to:

$$\sum_{i \in N'} \sum_{j \in N'} x_{ij} \leq \sum_{i \in N'} n_i - 1, \quad \forall N' \subset N, N' \neq \emptyset \tag{5.6}$$

The formulation presented in Sarin et al. (2011), denoted as HMATSP-SSY, is given as follows:

HMATSP-SSY: Minimize (5.1)

subject to :

(5.2) – (5.3), and (5.5)

$$\sum_{j \in N - \{1\}} p^u_{1j} = 1, \quad \forall u \in N - \{1\} \tag{5.7}$$

$$\sum_{j \in N,\ j \neq i,\ j \neq u} p^u_{ji} - \sum_{j \in N - \{1\},\ j \neq i} p^u_{ij} = 0, \quad \forall u \in N - \{1\}, \quad \forall i \in N - \{1\}, u \neq i \tag{5.8}$$

$$\sum_{j \in N,\ j \neq u} p^u_{ju} = 1, \quad \forall u \in N - \{1\} \tag{5.9}$$

$$p^u_{ij} \leq x_{ij}, \quad \forall u, j \in N - \{1\}, i \in N, i \neq j, i \neq u \tag{5.10}$$

$$p^u_{ij} \geq 0, \quad \forall u \in N - \{1\}, i \in N, j \in N - \{1\} \tag{5.11}$$

Constraints (5.7)-(5.10) represent subtour elimination constraints, which enforce connectivity of the graph induced by the **x**-variables by requiring a path defined by the flow variables p^u_{ij} to exist from city 1 to each city $u \in N - \{1\}$. Constraints (5.7) ensure that a unit of flow commences from the base for each city $u \in N - \{1\}$. Constraints (5.8) are the standard flow conservation constraints. Constraints (5.9) ensure that a unit of flow terminates at each city $u \in N - \{1\}$ (these are implied by (5.7) and (5.8) but we retain them for convenience). Constraints (5.10) restrict that a flow of one unit can occur from city i to city j only if they are connected. Constraints (5.11) assert logical requirements. We also introduce a scaling version of HMATSP-SSY, denoted by HMATSP-SSY$_{SC}$, which will be used later to establish relationship among HMATSP formulations, and it is as follows:

HMATSP-SSY$_{SC}$ Minimize (5.1)

subject to :

(5.2) – (5.3), (5.5), (5.8), and (5.11)

$$\sum_{j \in N-\{1\}} p^u_{1j} = n_i, \quad \forall u \in N - \{1\} \tag{5.12}$$

$$\sum_{j \in N,\ j \neq u} p^u_{ju} = n_i, \quad \forall u \in N - \{1\} \tag{5.13}$$

$$p^u_{ij} \leq n_u x_{ij}, \quad \forall u, j \in N - \{1\}, i \in N, i \neq j, i \neq u \tag{5.14}$$

Constraints (5.12) and (5.13) are identical to (5.7) and (5.9) except that n_i units of flow depart and return to the base city, and Constraints (5.14) are scaled versions of Constraints (5.10).

To begin our analysis, we first apply Benders decomposition principle to HMATSP-SSY in order to investigate its performance against a direct solution using CPLEX.

Benders decomposition-based approach for HMATSP-SSY

The Benders subproblem (BSP) comprises Constraints (5.7)-(5.11). Let β^u, π^u_{ij}, ρ^u, and γ^u_{ij} be dual variables associated with Constraints (5.7), (5.8), (5.9), and (5.10), respectively. Note that the subproblem is a feasibility problem that generates the following Benders feasibility cuts for inclusion in the master problem:

$$\sum_{u \in N-\{1\}} \sum_{\substack{i \in N \\ i \neq u}} \sum_{\substack{j \in N-\{1\} \\ j \neq i}} (-x_{ij}) \gamma^u_{ij} + \sum_{u \in N-\{1\}} \beta^u + \sum_{u \in N-\{1\}} \rho^u \leq 0. \tag{5.15}$$

$$\forall\ (\gamma^u_{ij},\ \pi^u_{ij},\ \beta^u,\ \rho^u, \forall i, j, u) \in \Omega$$

where Ω is the set of extreme rays of the polyhedron defined by the dual of the subproblem.

The Benders master problem (BMP) accordingly is given as follows:

BMP: Minimize (5.1); subject to: (5.2)-(5.3), (5.5), and (5.15)

5.2.2 A single commodity formulation for the HMATSP

We now propose a single-commodity flow-based formulation for the HMATSP, denoted as HMATSP-SCF, which turns out to be an aggregated version of HMATSP-SSY, but one that is computationally more competitive as demonstrated in Section 5.5 below. This formulation is also based on using a flow representation (in an aggregated fashion this time) in order to obviate subtours. Consider the following additional notation:

y_{ij} : integer variable that indicates the flow on arc (i, j) in an HMATSP solution, $i \neq j$, $i, j \in N$.

K : positive constant (given by $\sum_{i \in N-\{1\}} n_i$).

HMATSP-SCF : Minimize (5.1)

subject to

(5.2) – (5.3), and (5.5)

$$\sum_{j \in N-\{1\}} y_{1j} = \sum_{i \in N-\{1\}} n_i \tag{5.16}$$

$$y_{ij} \leq K x_{ij}, \quad \forall i, j \in N, j \neq i \tag{5.17}$$

$$\sum_{j \in N,\ j \neq i} y_{ji} - \sum_{j \in N-\{1\},\ j \neq i} y_{ij} = n_i, \quad \forall i \in N - \{1\} \tag{5.18}$$

$$y_{ij} \geq 0, \quad \forall i, j \in N, i \neq j. \tag{5.19}$$

Constraints (5.16)- (5.18) are subtour elimination constraints, and they impose connectivity of the graph induced by **x**-variables by requiring a path defined by the flow variables y_{ij} to exist from city 1 to each city $i \in N - \{1\}$ with n_i units flowing into each city i. Constraints (5.16) assure that $K = \sum_{i \in N-\{1\}} n_i$ units leave the depot. Constraints (5.17) assure that a flow can occur between cities i and j only if they are connected. Constraints (5.18) enforce that n_i units flow to city i, $\forall i \in N-\{1\}$. Observe that Constrains (5.18) imply (5.16) in that they enforce $\sum_{i \in N-\{1\}} n_i$ units to depart from the base (source) to satisfy the n_i units required for each city i, $i \in N - \{1\}$. However, we state these constraints explicitly in order to provide insights into the relationship of the above formulation to the HMATSP-SSY, as revealed in Proposition 5.2.1 below. Finally, Constraints

(5.19) specify domains of the decision variables. As Constraints (5.16)- (5.18) are standard flow conservation constraints and satisfy the unimodularity property, the **y**-variables take integer values at extreme point solutions for integral values of **x**. Note that, if we restrict $n_i = 1$, and let $x_{ii} = 0$, $\forall i \in N$, as in the case of the traveling salesman problem (TSP), then the above formulation reduces to that proposed by Gavish and Graves Gavish and Graves (1978) for the TSP.

Proposition 5.2.1. *HMATSP-SCF is a valid formulation for the HMATSP.*

Proof. We prove this result by showing equivalence between HMATSP-SCF and HMATSP-SSY. To this end, consider a digraph generated by **x**-variables for any feasible solution **(x,y)** to HMATSP-SCF. First of all, note that constraints (5.2) and (5.3) are included in both formulations. Constraints (5.16) enforce that $\sum_{i \in N-\{i\}} n_i$ units leave the base city, and constraints (5.17) ensure that a flow can occur only in those arcs (i, j), $i \in N, j \in N, i \neq j$, for which $x_{ij} > 0$. Furthermore, constraints (5.18) enforce flow balance, i.e, flow in minus flow out at each city u to be equal to n_u. In other words, there exits a path involving **x** variables for which n_u units flow from node 1 into each city $u \in N - \{1\}$. Let us denote a path for each $u \in N - \{1\}$ as T^u. Then, set $p^u_{ij} = 1$ for all $(i, j) \in T^u$, where $x_{ij} \geq 1$ for all $(i, j) \in T^u$, and let $p^u_{ij} = 0$ otherwise. This yields a flow from node 1 to node u that satisfies flow conservation constraints (5.7)-(5.10) of HMATSP-SSY. Therefore, there exits a feasible solution **(x,p)** to HMATSP-SSY having the same objective value.

Conversely, consider a feasible solution **(x,p)** to the HMATSP-SSY. Constraints (5.7)-(5.10) ensure a path from node 1 to each node u for all $u \in N - \{1\}$ denoted by T^u. Then, if we let $y_{ij} = \sum_{u \in N-\{1\}} n_u p^u_{ij}$, they satisfy constraints (5.16)- (5.18) of HMATSP-SCF. □

Let $z_{LP}(\text{SSY})$, $z_{LP}(\text{SSY}_{SC})$ and $z_{LP}(\text{SCF})$, respectively, denote the objective values of the linear programming relaxations of HMATSP-SSY, the scaling version of HMATSP-SSY, and HMATSP-SCF. Then, we have the following result:

Proposition 5.2.2. *$z_{LP}(SCF) \leq z_{LP}(SSY)$. Moreover, this inequality can be strict.*

Proof. HMATSP-SCF is an aggregated version of HMATSP-SSY_{SC}, which is evident by substituting $y_{ij} = \sum_{u \in N-1} n_u p^u_{ij}$, whereby (5.12) becomes (5.16), (5.14) becomes (5.17), and (5.8) and (5.13)

imply (5.18). Hence, $z_{LP}(\text{SCF}) \leq z_{LP}(\text{SSY}_{SC})$. It can be shown that $z_{LP}(\text{SSY}_{SC}) = z_{LP}(\text{SSY})$, and therefore, $z_{LP}(\text{SCF}) \leq z_{LP}(\text{SSY})$. To show that this last inequality can be strict, consider the instance presented in Table 5.3 that is denoted by ftv55, where $z_{LP}(\text{SSY}) = 5142$ and $z_{LP}(\text{SCF}) = 5102.96$. □

Benders decomposition-based approach for HMATSP-SCF

Similar to HMATSP-SSY, we can apply Benders decomposition to the formulation HMATSP-SCF. The Benders subproblem (BSP) comprises Constraints (5.16)-(5.19). However, we can omit Constraints (5.16) because, as noted above, they are implied by Constrains (5.18). Let α_{ij} and β_i be the dual variables associated with Constraints (5.17) and (5.18), respectively. As before, the BSP is just a feasibility problem, and therefore, only Benders feasibility cuts are generated, as given by:

$$\sum_{i \in N-\{1\}} n_i \beta_i - K \sum_{i \in N} \sum_{j \in N} x_{ij} \alpha_{ij} \leq 0. \tag{5.20}$$
$$\forall\ (\alpha_{ij},\ \beta_i, \forall i, j) \in \Omega'$$

where Ω' is the set of extreme rays of the polyhedron defined by the dual of the subproblem.

Following the notation introduced in Gavish and Graves (1978), let N_1 and N_2 be two sets in which the cities are divided, where $i \in N - \{1\}$ belongs to N_1 if city i is in the subtour containing the depot 1, and otherwise, $i \in N_2$. An extreme ray, $r \in \Omega'$, can be obtained as follows:

$$\beta_i = 0 \ \text{ if } \ i \in N_1 \ \text{ or } \ \beta_i = 1 \ \text{ if } \ i \in N_2 \tag{5.21}$$

$$\alpha_{ij} = \mid u_i - u_j \mid, \quad i, j \in N - \{1\}, i \neq j. \tag{5.22}$$

Thus, the Benders cut, (5.20), can be rewritten as

$$\sum_{i \in N_1} \sum_{j \in N_2} x_{ij} \geq \frac{\sum_{i \in N_2} n_i}{K}. \tag{5.23}$$

Since $\sum_{i \in N_2} n_i \leq K = \sum_{i \in N-\{1\}} n_i$, and x_{ij} is integer, the Benders cut can be rewritten as:

$$\sum_{i \in N_1} \sum_{j \in N_2} x_{ij} \geq 1. \tag{5.24}$$

Therefore, the Benders feasibility cuts (5.20) are equivalent to Constraints (5.4). The same results have been shown by Gavish and Graves (1978) for the case of the ATSP. This leads to the following

Benders master problem (BMP):

BMP: Minimize (5.1); subject to: (5.2)-(5.3), (5.5), and (5.24)

In a standard Benders decomposition approach as the one developed in Gavish and Graves (1978), the dual of the subproblem is solved by a linear programming solver, and every time a unique ray is identified, a unique Benders cut in the form of (5.20) or (5.24) is appended to the master problem. On the other hand, we propose to exploit structure of the primal subproblem, Constraints (5.16)-(5.19) enforce connectivity of the graph induced by the **x**-variables. We use this graphic representation to indentify subtours. This way of determining subtours is more efficient than solving the primal or the dual of the subproblem by a standard linear programming solver such as CPLEX to obtain a Benders feasibility cut, which can be very time consuming for large-sized problems. Besides, while the solution to a Benders subproblem can generate only one SEC, our approach can generate all SECs at the same time.

Given a x-solution (integer values) to the master problem satisfying Constraints (5.2)-(5.3) and (5.5), we identify all node subsets N_k, $k = 1, ..., s$, which form disjoint connected subgraphs or subtours, where the N_1 contains the base city $\{1\}$, and s represents the total number of such node sets in the current solution. To identify these subtours, we use the approach presented in Algorithm 1 and described below, but first, consider the following additional notation.

N_k	:	A subset of nodes forming subtour k, $k = 1, ..., s$.
N_{kp}	:	The city in position p in set k, $k = 1..., s$, $p = 1, ...,$.
h_k	:	The head (initial city) of a set S_k, $k = 1, ..., s$..
t_k	:	A temporary tail of an arc in S_k, $k = 1, ..., s$.
f	:	An indicator that equals 0 if no more cities in the solution are connected to set S_k.
$x_{[i][j]}$	:	The value of arc (i, j) in the current solution, $i, j \in N$.
n_a	:	The number of arcs in the current solution for which $x_{[i][j]}$ is greater than zero, $i, j \in N$, $i \neq j$.

Algorithm 1 starts by determining an initial city (head of the set forming the subtour), and then identifying all cities connected to this head in the current solution. The outputs of Algorithm 1 are

N_1,...,N_s sets, in which each city $i \in N$ belongs to a unique set. The steps of this algorithms are as follows. In Step 1, n_a, k, and f are initialized. In Step 2, the termination condition is checked. If the number of arcs in the current solution (n_a) is greater than zero, then Steps 3-30 are performed; otherwise, the algorithm stops. In steps 4-12, the head of N_k (h_k) forming the subtour is identified. In Steps 13-28 all cities connected to h_k are added to N_k. Note that in Step 22, every time a new city is added to N_k, Steps 13-30 are reinitialized. If N_k is not connected to another city, then k is increased by one (Step 29).

Algorithm 1 Approach to identify all subtours in the current integer HAMTSP solution.

1: $n_a = 1$, $k = 1$, $f = 0$
2: **while** $n_a > 0$ **do**
3: $n_a = 0$
4: **for** $i = 1$ **to** $|N|$ **do**
5: **for** $j = 1$ **to** $|N|$ **do**
6: **if** $i \neq j$ and $x_{i,j} > 0$ **then**
7: $n_a = n_a + 1$, $h = i$, and $t = j$
8: $N_k = \{i, j\}$ and $x_{[i][j]} = 0$
9: break
10: **end if**
11: **end for**
12: **end for**
13: **for** $i = 1$ **to** $|N_k|$ **do**
14: **for** $j = 1$ **to** $|n|$ **do**
15: **if** $N_{ki} \neq j$ and $x_{[N_k i][j]} > 0$ **then**
16: f=0
17: **for** $p = 1$ **to** $|S_k|$ **do**
18: **if** $N_{ki} \neq j$ and $x_{[N_{ki}][j]} > 0$ **then**
19: $f = f + 1$ and break
20: **if** $f = 0$ **then**
21: $N_k = N_k \cup j$ and $x_{[N_{ki}][j]} = 0$
22: i=0 and j=0
23: **end if**
24: **end if**
25: **end for**
26: **end if**
27: **end for**
28: **end for**
29: k=k+1
30: **end while**

The proposed approaches (described below) were implemented using CPLEX's lazy constraints callback function, where specific violated SECs (if any) are generated whenever an integer solution to the current relaxed problem is identified. These approaches are implemented in a traditional branch-and-cut scheme, where at each node SECs are checked whenever an integer-feasible solution has been identified. If this solution does not involve subtours, then we obtain an incumbent solution. Otherwise, SECs are added to the model, and the linear programming (LP) relaxation of the problem is solved. After adding SECs, we have three possibilities. (1) We obtain an integer solution containing no subtours, in which case, we have an incumbent solution. (2) We obtain an integer solution containing subtours, in which case we repeat the process, and (3) we obtain a non-integer solution in which case the solver continues solving the LP relaxation of the problem until an integer solution is reached. Thus, the procedure finally terminates with an optimal solution to the original problem by only enforcing SECs whenever they are violated. In this sense, the SECs are implemented as lazy cuts.

We test several approaches to accelerate the Benders decomposition scheme as described below:

Approach 1 (A_1): Append a single cut of type (5.24) based on the set N_1 as done in the standard Benders approach.

Approach 2 (A_2): Generate all s cuts of type (5.4) based on the sets N_1,..., N_s.

Approach 3 (A_3): Generate a single cut of type (5.6) based on the set having the largest cardinality from N_1,..., N_s.

Approach 4 (A_4): Generate all s cuts of type (5.6) based on the sets N_1,..., N_s.

The accelerated Benders methods ($A_1 - A_4$) can be viewed as cutting plane algorithms (row-generation implementation), where solutions violating Constraints (5.4) or (5.6) are identified (separated) using Algorithm 1 only when variables x_{ij} take integer values (or that subtour elimination constraints (SECs) are generated from integer points).

Finally, we add the single-city subtour elimination constraints to all formulations and Benders

algorithms developed for the HMATSP and its extensions, which are given by:

$$x_{ii} \leq n_i - 1, \quad \forall i \in N - \{1\}. \tag{5.25}$$

5.3 Transformation of HMATSP to an equivalent ATSP representation

We also investigate the performance of a straightforward approach of solving the HMATSP by transforming to an equivalent ATSP representation with the aim of using an effective formulation developed in the literature for the ATSP. This transformation can be achieved by copying each city, $i \in N - \{1\}$, n_i times and modifying the distance matrix $[c_{ij}]$ accordingly. Thus, the number of cities in the equivalent ATSP representation is $\sum_{\in N-\{1\}} n_i + 1$. Next, we present the most effective compact formulations for ATSP for solving directly by CPLEX.

Recently in Roberti and Toth (2012), several compact formulations for the ATSP have been compared, and it has been shown that the three most effective formulations from the viewpoint of a direct solution by CPLEX are the GG (Gavish and Graves (1978)), MTZ (Miller et al. (1960)), and DL (Desrochers and Laporte (1991)), which are presented next and will be used as a benchmark for comparison with the compact formulation presented in this chapter for the HMATSP.

Letting y_{ij} be a real variable indicating the flow in arc (i, j), then the GG-SECs are given by:

$$y_{ij} \leq (|N| - 1)x_{ij}, \quad i, j \in N, \quad i \neq j \tag{5.26}$$

$$\sum_{j \in N,\ j \neq i} y_{ji} - \sum_{j \in N',\ j \neq i} y_{ij} = 1, \quad \forall i \in N - \{1\} \tag{5.27}$$

$$y_{ij} \geq 0, \quad i, j \in N, \quad i \neq j. \tag{5.28}$$

Letting u_i indicate a real number representing the order in which city i is visited in an optimal

tour, $i \in N - \{1\}$, then the MTZ-SECs can be stated as follows:

$$u_i - u_j + (|N| - 1)x_{ij} \leq |N| - 2, \quad i, j \in N - \{1\}, \quad i \neq j \tag{5.29}$$

$$1 \leq u_i \leq |N| - 1, \quad i \in N - \{1\}. \tag{5.30}$$

The MTZ-SECs were strengthen by Desrochers and Laporte (1991), and we designate them as DL-SECs which are as follows:

$$u_i - u_j + (|N| - 1)x_{ij} + (|N| - 3)x_{ij} \leq |N| - 2, \quad i, j \in N - \{1\}, \quad i \neq j \tag{5.31}$$

$$-u_i + (|N| - 3)x_{i1} + \sum_{j \in N-\{1\},\ j \neq i} x_{ji} \leq -1, \quad i \in N - \{1\} \tag{5.32}$$

$$u_i + (|N| - 3)x_{1i} + \sum_{j \in N-\{1\},\ j \neq i} x_{ij} \leq |N| - 1, \quad i \in N - \{1\}. \tag{5.33}$$

Finally, we also add the two-city subtour elimination constraints to all ATSP formulations, which are given by:

$$x_{ij} + x_{ji} \leq 1, \quad \forall i, j \in N - \{1\},\ i \neq j. \tag{5.34}$$

5.4 Extensions of the HMATSP

In this section, we adapt the single-commodity flow-based formulation for some extensions of the HMATSP, and develop accelerated Benders algorithms for their solution.

5.4.1 Single-base multiple salesmen HMATSP

The single-base high-multiplicity multiple traveling salesmen problem (SB-HMmATSP) is a natural extension of the HMATSP wherein m salesmen are located at a single base and can visit cities multiple times. Formally, this problem can be stated as follows:

Given a complete graph with vertex set N, an asymmetric distance matrix $[c_{ij}]$*, and positive integers*

n_i, $\forall i \in N$, find m tours that start and end at the given base after having visited node i exactly n_i times, $\forall i \in N$, such that the total distance traveled is minimized.

The SB-HMmATSP can be modeled as follows:

SB-HMmATSP : Minimize (5.1)

subject to

$$\sum_{j \in N} x_{1j} = m, \tag{5.35}$$

$$\sum_{j \in N} x_{j1} = m, \tag{5.36}$$

$$\sum_{j \in N} x_{ji} = n_i, \quad \forall i \in N - \{1\} \tag{5.37}$$

$$\sum_{j \in N} x_{ij} = n_i, \quad \forall i \in N - \{1\} \tag{5.38}$$

$$x_{ij} \geq 0 \text{ and integer}, \quad \forall i, j \in N, \tag{5.39}$$

SECs. (5.40)

Constraints (5.35) and (5.36) enforce m salesmen to depart from and return to the base city, and Constraints (5.37) and (5.38) enforce city i, $i \in N - \{1\}$ to be visited and departed from n_i times. Constraints (5.39) specify the domain of decision variables, and Constraints (5.40) are the subtour elimination constraints (SECs) which are discussed next.

It can be shown that Constraints (5.6) are not valid for the SB-HMmATSP, and they need some refinement. Consider an instance having five cities ($n = 5$), two salesmen ($m = 2$), and $n_1 = 1$, $n_2 = 2$, $n_3 = 1$, $n_4 = 1$, and $n_5 = 1$ with a valid solution given by route 1: $x_{12} = 1$, $x_{22} = 1$, $x_{23} = 1$, $x_{31} = 1$, and route 2: $x_{14} = 1$, $x_{45} = 1$, $x_{51} = 1$. If we let $N' = \{1, 2, 3\}$, then Constraints (5.6) impose $x_{11} + x_{12} + x_{13} + x_{21} + x_{22} + x_{23} + x_{31} + x_{32} + x_{33} \leq 3$, which cuts off this feasible solution. Therefore, (5.6) needs to be imposed just for subsets of cities not involving city 1 as follows:

$$\sum_{i\in N'}\sum_{j\in N'} x_{ij} \leq \sum_{i\in N'} n_i - 1, \quad \forall N' \subset N - \{1\}, N' \neq \emptyset \tag{5.41}$$

We can derive the following exponential-length formulations for the SB-HMmATSP

SB-HMmATSP1: Minimize (5.1); subject to: (5.35)- (5.39) and (5.4).

SB-HMmATSP2: Minimize (5.1); subject to: (5.35)- (5.39), and (5.41).

Polynomial-length SECs for the SB-HMmATSP are as follows:

$$y_{ij} \leq K' x_{ij}, \quad \forall i, j \in N, i \neq j \tag{5.42}$$

$$\sum_{j\in N,\ j\neq i} y_{ji} - \sum_{j\in N-\{1\},\ j\neq i} y_{ij} = n_i, \quad \forall i \in N - \{1\} \tag{5.43}$$

$$y_{ij} \geq 0, \quad \forall i, j \in N, i \neq j, \tag{5.44}$$

where $K' = \sum_{i\in N-\{1\}} n_i - (m-1)$.

Therefore, we propose the following compact formulation for the SB-HMmATSP:

SB-HMmATSP3: Minimize (5.1); subject to: (5.35)- (5.39), and (5.42)-(5.44).

Benders decomposition-based approach for the solution of SB-HMmATSP

We also apply Benders decomposition to SB-HMmATSP3 for the solution of SB-HMmATSP, where the Benders subproblem (BSP) is given by Constraints (5.42)-(5.44). Let α'_{ij} and β'_i be the dual variables associated with Constraints (5.42)-(5.43), respectively, Benders feasibility cuts are generated as follows:

$$\sum_{i\in N-\{1\}} \beta'_i - K' \sum_{i\in N}\sum_{j\in N} x_{ij}\alpha'_{ij} \leq 0, \quad \forall\ (\alpha'_{ij},\ \beta'_i, \forall i, j) \in \bar{\Omega}, \tag{5.45}$$

where $\in \bar{\Omega}$ is the set of extreme rays of the polyhedron defined by the dual of the subproblem.

Again, let N_1 and N_2 be two sets in which the cities are divided, where $i \in N - \{1\}$ belongs to N_1 if city i is in the subtour containing the depot (1), and $i \in N_2$, otherwise. An extreme ray, $r \in \bar{\Omega}$,

can be obtained as follows:

$$\beta_i = 0 \text{ if } i \in N_1 \text{ or } \beta_i = 1 \text{ if } i \in N_2 \tag{5.46}$$

$$\alpha_{ij} = \mid u_i - u_j \mid, \quad i, j \in N - \{1\}, i \neq j. \tag{5.47}$$

Thus, the Benders cut can be rewritten as

$$\sum_{i \in N_1} \sum_{j \in N_2} x_{ij} \geq \frac{\sum_{i \in N_2} n_i}{K'} \tag{5.48}$$

Since $\sum_{i \in N_2} n_i \leq K'$, and x_{ij} is integer, the Benders feasibility cuts (5.45) can be strengthened by

$$\sum_{i \in N_1} \sum_{j \in N_2} x_{ij} \geq 1 \tag{5.49}$$

The Benders master problem is defined by:

BMP: Minimize (5.1); subject to: (5.35)- (5.39) and (5.49).

Similar to HMATSP, we propose several approaches to accelerate this Benders decomposition scheme. For a given x-solution (integer values) to the master problem satisfying Constraints (5.35)-(5.39), we identify all node subsets N_k, $k = 1, ..., s$, which form disjoint connected subgraphs or subtours, where the set N_1 contains the base city $\{1\}$, and s represents the total number of such node sets in the current solution. We explore several approaches as described below.

Approach 5 (A_5): Append a single cut of type (5.49) based on the set N_1 as done in the standard Benders approach.

Approach 6 (A_6): Generate s cuts of type (5.4) base on the sets N_1,..., N_s.

Approach 7 (A_7): Generate a single cut of type (5.41) based on the set having the largest cardinality from N_2,..., N_s.

Approach 8 (A_8): Generate $s - 1$ cuts of type (5.41) based on the sets N_2,..., N_s.

5.4.2 Non-fixed destination multiple base and multiple salesmen HMATSP

The non-fixed destination, multiple base, high-multiplicity, multiple traveling salesmen problem (NFMB-HMmATSP) is another extension of the HMATSP, which is stated as follows:

Given a complete graph with vertex set $N = B \cup N'$, where the first $|B|$ nodes of N comprise a set of base cities having m_b salesmen located initially at base $b \in B$, and where $N' = \{|B|+1, |B|+2, ..., n\}$ comprise the set of cities to be visited, and given an asymmetric distance matrix $[c_{ij}]$, and positive integers n_i, $\forall i \in N$, find $m = \sum_{b \in B} m_b$ tours, with m_b tours starting at the base city $b \in B$, and likewise, with m_b tours ending at a base city $b \in B$, where any such tour can start and end at different base cities while collectively having visited city i exactly n_i times, $\forall i \in N$, so that the total distance traveled is minimized.

Note that in this problem, the salesmen are not required to return to their respectively originating bases, but m_b salesmen must start at base b and likewise return to base b, $\forall b \in B$. An example of such a problem might be a vehicle routing problem where the salesmen represent identical vehicles of a given type and where the solution is to be repeated cyclically.

The NFMB-HMmATSP can be modeled as follows:

NFMB-HMmATSP: Minimize (5.1)

subject to

$$\sum_{i \in N'} x_{ib} = m_b, \quad \forall b \in B \tag{5.50}$$

$$\sum_{i \in N'} x_{bi} = m_b, \quad \forall b \in B \tag{5.51}$$

$$\sum_{j \in N} x_{bq} = 0, \quad \forall b, q \in B \tag{5.52}$$

$$\sum_{j \in N} x_{ij} = n_i, \quad \forall i \in N' \tag{5.53}$$

$$\sum_{j \in N} x_{ji} = n_i, \quad \forall i \in N' \tag{5.54}$$

$$x_{ij} \geq 0 \text{ and integer}, \quad \forall i, j \in N \tag{5.55}$$

$$SECs \tag{5.56}$$

Constraints (5.50) and (5.51) require that m_b salesmen depart from and return to each base $b \in B$. Constraints (5.52) enforce that the bases are not connected by restricting $x_{bq} = 0$, and Constraints (5.53)- (5.55) are identical to those for HMATSP-SCF. Constraints (5.56) are the subtour elimination constraints (SECs) which are discussed next.

It can be shown that Constraints (5.4) and (5.6) are not valid for the NFMB-HMmATSP, and they need some refinement. Consider an instance having two bases ($B = 2$) denoted by cities 1 and 2 each having one salesman ($m_1 = n_1 = 1$ and $m_2 = n_2 = 1$), and three cities denoted by 3, 4, and 5 with $n_3 = 2$, $n_4 = 1$, and $n_5 = 1$. Consider the valid solution given by route 1: $x_{13} = 1$, $x_{34} = 1$, $x_{43} = 1$, $x_{31} = 1$, and route 2: $x_{25} = 1$, and $x_{52} = 1$. If we let $N' = \{1, 3, 4\}$, then Constraints (5.4) impose $x_{12} + x_{15} + x_{32} + x_{35} + x_{42} + x_{45} \geq 1$, which cuts off this feasible solution. Constraints (5.6) impose $x_{11} + x_{13} + x_{14} + x_{31} + x_{33} + x_{34} + x_{41} + x_{43} + x_{44} \leq 3$, which also cuts off this feasible solution. Therefore, (5.4) and (5.6) needs to be imposed just for subsets of cities not involving any base B as follows:

$$\sum_{i \in N'} \sum_{j \in N - N'} x_{ij} \geq 1, \quad \forall N' \subset N - \{b \in B\}, N' \neq \emptyset \tag{5.57}$$

$$\sum_{i \in N'} \sum_{j \in N'} x_{ij} \leq \sum_{i \in N'} n_i - 1, \quad \forall N' \subset N - \{b \in B\}, N' \neq \emptyset. \tag{5.58}$$

Proposition 5.4.1. *SECs* (5.57) *and* (5.58) *are valid for the NFMB-HMmATSP*

Proof. If a set N', $N' \subset N - \{b \in B\}, N' \neq \emptyset$, is not connected to its complement $N - N'$, then $\sum_{i \in N'} \sum_{j \in N - N'} x_{ij} = 0$. Therefore (5.57) assures connectivity of N' with $N - N'$.

For any of the m proper tours of an NFMB-HMmATSP solution, the right-hand side of SECs (5.58) takes a value less or equal to $\sum_{i \in N'} n_i - 1$, and if the set N' is disconnected from its complement $(N - N')$, then the left-hand side takes a value greater than $\sum_{i \in N'} n_i - 1$ and that tour will not be permitted. □

Exponential-length formulation for the NFMB-HMmATSP are as follows:

NFMB-HMmATSP1: Minimize (5.1); subject to: (5.50)-(5.55), and (5.57).

NFMB-HMmATSP2: Minimize (5.1); subject to: (5.50)-(5.55), and (5.58).

Polynomial-length SECs for the NFMB-HMmATSP are as follows:

$$y_{ij} \leq K' x_{ij}, \quad \forall i, j \in N \tag{5.59}$$

$$\sum_{j \in N,\ j \neq i} y_{ji} - \sum_{j \in N - \{b \in B\},\ j \neq i} y_{ij} = n_i, \quad \forall i \in N' \tag{5.60}$$

$$y_{ij} \geq 0, \quad \forall i, j \in N, i \neq j. \tag{5.61}$$

We propose the following compact formulation for the NFMB-HMmATSP

NFMB-HMmATSP3: Minimize (5.1); subject to: (5.50)-(5.55), and (5.59)-(5.61).

Benders decomposition-based approach for the solution of NFMB-HMmATSP

The Benders subproblem (BSP) is given by Constraints (5.59)-(5.61). Let $\bar{\alpha}_{ij}$ and $\bar{\beta}_i$ be the dual variables associated with Constraints (5.59)-(5.60), respectively, Benders feasibility cuts are generated as follows:

$$\sum_{i \in N - \{1\}} \bar{\beta}_i - K \sum_{i \in N} \sum_{j \in N} x_{ij} \bar{\alpha}_{ij} \leq 0. \ \ \forall\ (\bar{\alpha}_{ij},\ \bar{\beta}_i, \forall i, j) \in \hat{\Omega}, \tag{5.62}$$

where $\hat{\Omega}$ is the set of extreme rays of the polyhedron defined by the dual of the subproblem.

The Benders master problem is defined by:

BMP: Minimize (5.1); subject to: (5.50)-(5.55), and (5.62).

Similar to HMATSP, we propose several approaches to accelerate this Benders decomposition scheme. For a given x-solution to the master problem satisfying Constraints (5.50)-(5.55), we identify all node subsets N_k, $k = 1, ..., s$, which form disjoint connected subgraphs or subtours, where s represents the total number of such nodes in the current solution. We explore several approaches as described below.

Approach 9 (A_9): Append a single cut of type (5.57) based on the set having the largest cardinality from N_1,...,N_s.

Approach 10 (A_{10}): Generate s cuts of type (5.57) based on sets N_1,...,N_s.

Approach 11 (A_{11}): Generate a single cut of type (5.58) based on the set having the largest cardinality from N_1,...,N_s.

Approach 12 (A_{12}): Generate s cuts of type (5.58) based on sets N_1,...,N_s.

5.5 Computational Results

As a benchmark, the different model formulations for the HMATSP as well as its extensions considered in this chapter, were initially solved directly using CPLEX 12.5.1 with OPL. Furthermore, the models HMATPS-SSY, HMATPS-SCF, SB-HMmASTSP, and MBNF-HMATSP were solved using Benders decomposition, where these algorithms were coded using Microsoft Visual Studio Professional 2012, with CPLEX as the solver for the master and dual subproblems. All runs were made on an Intel Xeon E5-2687W 3.5GHz computer having 32 GB of RAM running Windows 7 with a single thread, and the maximum permissible limit on CPU time was set to 3600 seconds.

Table 5.1 displays the ATSP dataset used in our experimentation to derive HMATSP instances. The columns indicate the instance name (name), the number of cities (n), and the source from where the instance was taken (source), respectively. We use 18 problems from the TSPLIB (1997) containing between 43-443 cities, and 15 instances reported in Cirasella et al. (2001) involving between 101-1001 cities, and 4 problems used in Roberti and Toth (2012) having between 108-200 cities. To generate corresponding instances for the HMATSP and its extensions, the number of required visits to each city, $n_i,\ i \in N$, was generated randomly using discrete uniform distributions U[1, 5] and U[6, 10].

5.5.1 HMATSP results

Results on the transformation of an HMATSP to an equivalent ATSP representation

We test the approach described in Section 5.3, where HMATSP is transformed to an equivalent ATSP representation by copying each city i n_i times, $i \in N - \{1\}$. Tables 5.2 presents results obtained for solving HMATSP and ATSP formulations directly by CPLEX. The first three columns of this table contain the problem name, the number of cities (n), and the distribution used to generate the number of visits to each city (n_i), respectively. The fourth column (n') indicates the number of cities resulting from the transformation of an HMATSP instance to an equivalent

Table 5.1: Well-known ATSP problems used to generate instances for the HMATSP and its extensions.

Name	*n*	*Source*
p43	43	TSPLIB (1997)
ftv44	45	TSPLIB (1997)
ftv47	47	TSPLIB (1997)
ry48	48	TSPLIB (1997)
ft53	53	TSPLIB (1997)
ftv55	56	TSPLIB (1997)
ftv64	65	TSPLIB (1997)
ft70	70	TSPLIB (1997)
ftv70	71	TSPLIB (1997)
kro124	100	TSPLIB (1997)
td100.1	101	Cirasella et al. (2001)
Balas108	108	Roberti and Toth (2012)
dc112	112	Cirasella et al. (2001)
Balas120	120	Roberti and Toth (2012)
dc134	134	Cirasella et al. (2001)
ftv140	141	TSPLIB (1997)
ftv150	151	TSPLIB (1997)
Balas160	160	Roberti and Toth (2012)
ftv160	161	TSPLIB (1997)
ftv170	171	TSPLIB (1997)
dc176	176	Cirasella et al. (2001)
dc188	188	Cirasella et al. (2001)
code198	198	Cirasella et al. (2001)
Balas200	200	Roberti and Toth (2012)
code253	253	Cirasella et al. (2001)
td316.10	317	Cirasella et al. (2001)
rbg323	323	TSPLIB (1997)
rbg358	358	TSPLIB (1997)
rbg403	403	TSPLIB (1997)
rbg443	443	TSPLIB (1997)
dc563	563	Cirasella et al. (2001)
atex8	600	Cirasella et al. (2001)
big702	702	Cirasella et al. (2001)
dc849	849	Cirasella et al. (2001)
dc895	895	Cirasella et al. (2001)
dc932	932	Cirasella et al. (2001)
td1000.20	1001	Cirasella et al. (2001)

ATSP representation, where $n' = \sum_{i \in N-\{1\}} n_i + 1$. The fifth column indicates the method in consideration. In particular, $\text{SSY}_{\text{HMATSP}}$ and $\text{SCF}_{\text{HMATSP}}$ represent the HMATSP formulations of Sarin et al. (2011) and the formulation proposed in this chapter, respectively, and GG_{ATSP}, MTZ_{ATSP}, and DL_{ATSP} denote the formulations for the asymmetric traveling salesmen problem (ATSP) proposed in Gavish and Graves (1978), Miller et al. (1960), and Desrochers and Laporte (1991), respectively. The last column denotes the time required to reach optimality (T), where *tl* denotes that a formulation fails to solve the problem to optimality in the allowed CPU time of 3600 seconds. Note that the results are depicted for three instances involving 45, 100, and 170 cities for which we generate n_i values using discrete uniform distributions $[1, 5]$ and $[6, 10]$. $\text{SCF}_{\text{HMATSP}}$ solved all these instances to optimality within 131.07 seconds, while $\text{SSY}_{\text{HMATSP}}$ only solved 4 out of the 6 cases to optimality. On the other hand, GG_{ATSP}, MTZ_{ATSP}, and DL_{ATSP} each solved to optimality only 2 out of the 6 cases. Therefore, solving HMATSP by transforming it to an equivalent ATSP representation is not very effective. The number of arcs for an HMATSP instance is $|N|\ x\ |N|$, while that for the transformed ATSP representation is $(\sum_{i \in N-\{1\}} n_i) + 1)^2$. For example, consider the instance denoted by ftv171 with n_i in $[6, 10]$, where the number of arcs (cities) increases from $29,241$ to $1,811,716$ (171 to $1,346$), which show a significant increment. We have highlighted in bold the minimum T/gap for each instance (in this and the subsequent tables).

Results on the comparison of compact formulations for the HMATSP

Tables 5.3 presents results obtained for solving $\text{SCF}_{\text{HMATSP}}$ and $\text{SSY}_{\text{HMATSP}}$ using CPLEX on 36 instances derived from the TSPLIB (1997) having 43-443 cities. The first three columns are identical to those in Table 5.2. Columns 4-6 present information on the optimality gap for the LP relaxation (Lp gap) computed as $100 * |(Z_{IP} - Z_{LP})/Z_{IP}|\%$, the best integer solution obtained (Z_{IP}), and the total CPU time in seconds or the optimality gap (gap) computed as $100 * |(Z_{IP} - Z_{LB})/Z_{IP}|\%$ (T/gap) obtained by $\text{SCF}_{\text{HMATSP}}$. Columns 7-9 are identical to columns 4-6 and they present the results obtained for $\text{SSY}_{\text{HMATSP}}$. The symbols "-" and "tl" denote that a feasible solution could not be obtained for that instance in the prescribed time limit (tl=3600 seconds). If a feasible solution

Table 5.2: Results obtained by solving HMATSP instances by formulations developed for this problem (SSY_{HMATSP} and SCF_{HMATSP}), and by ATSP formulations (GG_{ATSP}, DL_{ATSP}, MTZ_{ATSP}). For the latter case each instance of the HMATSP is transformed into an equivalent ATSP representation by copying each city i, $i \in N - \{1\}$, n_i times.

Problem	n	n_i	n'	*Method*	T
ftv44	45	U[1,5]	-	SSY_{HMATSP}	7.25
			-	SCF_{HMATSP}	**0.17**
			129	GG_{ATSP}	2.26
			129	DL_{ATSP}	10.73
			129	MTZ_{ATSP}	4.85
		U[6,10]	-	SSY_{HMATSP}	5.85
			-	SCF_{HMATSP}	**0.09**
			348	GG_{ATSP}	63.62
			348	DL_{ATSP}	1373.06
			348	MTZ_{ATSP}	2734.35
kro124	100	U[1,5]	-	SSY_{HMATSP}	1468.58
			-	SCF_{HMATSP}	**45.33**
			272	GG_{ATSP}	tl
			272	DL_{ATSP}	tl
			272	MTZ_{ATSP}	tl
		U[6,10]	-	SSY_{HMATSP}	1092.44
			-	SCF_{HMATSP}	**11.95**
			807	GG_{ATSP}	tl
			807	DL_{ATSP}	tl
			807	MTZ_{ATSP}	tl
ftv170	171	U[1,5]	-	SSY_{HMATSP}	tl
			-	SCF_{HMATSP}	**131.07**
			518	GG_{ATSP}	tl
			518	DL_{ATSP}	tl
			518	MTZ_{ATSP}	tl
		U[6,10]	-	SSY_{HMATSP}	tl
			-	SCF_{HMATSP}	**24.37**
			1,346	GG_{ATSP}	tl
			1,346	DL_{ATSP}	tl
			1,346	MTZ_{ATSP}	tl

was obtained within the time limit, then we report the gap. $\text{SCF}_{\text{HMATSP}}$ attained optimality for all the instances within the specified time limit except for both the cases derived from p43 (see the first and second rows in Table 5.3), while $\text{SSY}_{\text{HMATSP}}$ obtained optimal solutions for instances containing up to 100 cities except for both the cases derived from p43, for which it was unable to generate any feasible integer solution. It is worth mentioning that the proposed formulation, $\text{SCF}_{\text{HMATSP}}$, solved instances to optimality containing between 100-443 cities, for which $\text{SSY}_{\text{HMATSP}}$ was unable to generate a feasible solution.

Tables 5.4 presents results obtained for solving $\text{SCF}_{\text{HMATSP}}$ and $\text{SSY}_{\text{HMATSP}}$ using CPLEX on 36 instances derived from the problems presented in Roberti and Toth (2012) and Cirasella et al. (2001) having 101-1001 cities. $\text{SCF}_{\text{HMATSP}}$ solved to optimality 24 out of the 36 instances, while $\text{SSY}_{\text{HMATSP}}$ solved only 1 out of the 36 instances to optimality. Therefore, $\text{SSY}_{\text{HMATSP}}$ clearly outperformed $\text{SSY}_{\text{HMATSP}}$; however, the former failed to generate a feasible solution for 12 instances containing between 563-1001 cities (see the last twelve rows in Table 5.4).

Based on the previous results, we can conclude the following: 1. As expected, $\text{SCF}_{\text{HMATSP}}$ was not as tight as $\text{SSY}_{\text{HMATSP}}$, but it was more efficient when solved directly by CPLEX 12.5.1. 2. $\text{SCF}_{\text{HMATSP}}$ could solve instances (derived from well-known ATSP libraries) having up to 443 cities within one hour of CPU time, while $\text{SSY}_{\text{HMATSP}}$ could only solve instances having up to 120 cities. 3. The main reason for $\text{SCF}_{\text{HMATSP}}$ to be faster than $\text{SSY}_{\text{HMATSP}}$ is a lower number of variables used to avoid subtours. Table 5.5 presents a comparison between the size of each formulation for some HMATSP instances. Columns 1-3 are identical to those described in the previous tables. Columns 4-6 represent the number of integer variables (x_{ij} Vars.), the number of variables used to avoid subtpours (SECs Vars), and the total number of variables (Total Vars.) employed for each formulation. For example, consider the instance denoted by rgb443 with 443 cities, where the total number of variables used by $\text{SSY}_{\text{HMATSP}}$ and $\text{SCF}_{\text{HMATSP}}$ are $86,742,501$ and $391,613$, respectively.

Table 5.3: Results obtained by solving HMATSP formulations (SCF_{HMATSP} and SSY_{HMATSP}) directly using CPLEX 12.5.1 on 36 instances derived from the TSPLIB (1997) having 43-443 cities.

			SCF_{HMATSP}			SSY_{HMATSP}		
Problem	n	n_i	*Lp gap*	Z_{IP}	*T/gap*	*LP gap*	Z_{IP}	*T/gap*
p43	43	U[1,5]	88.10%	5623	**0.05%**	-	-	tl
		U[6,10]	86.88%	5623	**0.05%**	-	-	tl
ftv44	45	U[1,5]	0.00%	5194	**0.17**	0.00%	5194	7.25
		U[6,10]	0.00%	12506	**0.09**	0.00%	5194	5.85
ftv47	48	U[1,5]	0.49%	5764	**0.89**	0.00%	5764	10.84
		U[6,10]	0.08%	13608	**0.14**	0.00%	13608	4.71
ry48	48	U[1,5]	1.98%	41644	**6.65**	0.00%	41644	21.53
		U[6,10]	0.21%	102829	**0.23**	0.00%	102829	12.87
ft53	53	U[1,5]	2.90%	21077	12.53	0.00%	21077	**12.29**
		U[6,10]	0.85%	51652	**0.28**	0.00%	51652	7.72
ftv55	56	U[1,5]	0.76%	5142	**0.80**	0.00%	5142	29.23
		U[6,10]	0.44%	11937	**1.17**	0.00%	11937	50.48
ftv64	65	U[1,5]	0.00%	5461	**0.39**	0.00%	5461	37.27
		U[6,10]	0.01%	14770	**0.55**	0.00%	14770	60.11
ft70	70	U[1,5]	0.15%	1119659	**1.33**	0.00%	119659	69.31
		U[6,10]	0.05%	308140	**0.69**	0.00%	308140	37.69
ftv70	71	U[1,5]	0.08%	6275	**1.12**	0.00%	6275	85.18
		U[6,10]	0.29%	15602	**0.95**	0.00%	15602	98.56
kro124	100	U[1,5]	0.64%	103840	**45.33**	0.00%	103840	1468.58
		U[6,10]	0.02%	277397	**11.95**	0.00%	277397	1092.44
ftv140	141	U[1,5]	0.15%	7831	**17.22**	-	-	tl
		U[6,10]	0.01%	19329	**13.99**	-	-	tl
ftv150	151	U[1,5]	0.24%	8625	**37.30**	-	-	tl
		U[6,10]	0.00%	21627	**11.81**	-	-	tl
ftv160	161	U[1,5]	0.00%	9331	**15.74**	-	-	tl
		U[6,10]	0.01%	21664	**18.38**	-	-	tl
ftv170	171	U[1,5]	0.11%	9422	**131.07**	-	-	tl
		U[6,10]	0.00%	22151	**24.37**	-	-	tl
rgb323	323	U[1,5]	0.00%	894	**873.06**	-	-	tl
		U[6,10]	0.00%	788	**442.28**	-	-	tl
rgb358	358	U[1,5]	0.00%	690	**1320.44**	-	-	tl
		U[6,10]	0.00%	533	**712.13**	-	-	tl
rgb403	403	U[1,5]	0.00%	845	**2042.04**	-	-	tl
		U[6,10]	0.00%	531	**893.56**	-	-	tl
rgb443	443	U[1,5]	0.00%	956	**1766.93**	-	-	tl
		U[6,10]	0.00%	570	**1075.31**	-	-	tl

Table 5.4: Results obtained by solving HMATSP formulations (SCF_{HMATSP} and SSY_{HMATSP}) directly using CPLEX 12.5.1 on 36 instances derived from Roberti and Toth (2012) and Cirasella et al. (2001) having 101-1001 cities.

			SCF_{HMATSP}			SSY_{HMATSP}		
Problem	n	n_i	*Lp gap*	Z_{IP}	*T/gap*	*LP gap*	Z_{IP}	*T/gap*
td100.1	101	U[1,5]	0.00%	31916	**4.18**	-	-	tl
		U[6,10]	0.01%	81709	**4.32**	-	-	tl
Balas108	108	U[1,5]	1.92%	472	**56.57**	-	-	tl
		U[6,10]	0.86%	1046	**13.48**	-	-	tl
dc112	112	U[1,5]	0.76%	11126	**0.53%**	-	-	tl
		U[6,10]	0.81%	11132	**0.57%**	-	-	tl
Balas120	120	U[1,5]	1.34%	916	**98.09**	0.00%	40907	97.06%
		U[6,10]	0.41%	2062	**30.50**	0.00%	2062	3552.61
dc134	134	U[1,5]	0.76%	5641	**0.67%**	-	-	tl
		U[6,10]	0.41%	5621	**0.32%**	-	-	tl
Balas160	160	U[1,5]	0.19%	1400	**60.22**	-	-	tl
		U[6,10]	0.16%	2917	**15.91**	-	-	tl
dc176	176	U[1,5]	1.28%	8644	**0.97%**	-	-	tl
		U[6,10]	2.72%	8773	**2.41%**	-	-	tl
dc188	188	U[1,5]	2.75%	13239	**23.31%**	-	-	tl
		U[6,10]	2.34%	11244	**9.59%**	-	-	tl
Code198	198	U[1,5]	0.00%	45058	**88.83**	-	-	tl
		U[6,10]	0.00%	9E+09	**50.29**	-	-	tl
Balas200	200	U[1,5]	0.93%	1320	**1861.58**	-	-	tl
		U[6,10]	0.47%	2942	**409.36**	-	-	tl
code253	253	U[1,5]	0.00%	4E+09	**265.37**	-	-	tl
		U[6,10]	0.00%	8E+09	**134.18**	-	-	tl
td316	317	U[1,5])	0.00%	2279351	**419.42**	-	-	tl
		U[6,10]	0.00%	5701444	**329.91**	-	-	tl
dc563	563	U[1,5]	-	-	tl	-	-	tl
		U[6,10]	-	-	tl	-	-	tl
atex8	600	U[1,5]	-	-	tl	-	-	tl
		U[6,10]	-	-	tl	-	-	tl
big702	702	U[1,5]	-	-	tl	-	-	tl
		U[6,10]	-	-	tl	-	-	tl
dc895	895	U[1,5]	-	-	tl	-	-	tl
		U[6,10]	-	-	tl	-	-	tl
dc932	932	U[1,5]	-	-	tl	-	-	tl
		U[6,10]	-	-	tl	-	-	tl
td1000	1001	U[1,5]	-	-	tl	-	-	tl
		U[6,10]	-	-	tl	-	-	tl

Table 5.5: Comparison between the total number of variables required by $\text{SSY}_{\text{HMATSP}}$ and $\text{SCF}_{\text{HMATSP}}$ to model the HMATSP using 4 instances derived from TSPLIB (1997) having 43-443 cities.

Problem	*n*	*Method*	*# x_{ij} Vars*	*# Subtour Vars.*	*Total Vars.*
p43	43	$\text{SSY}_{\text{HMATSP}}$	1,849	75,852	77,701
		$\text{SCF}_{\text{HMATSP}}$	1,849	1,764	3,613
ftv70	71	$\text{SSY}_{\text{HMATSP}}$	5,041	347,900	352,941
		$\text{SCF}_{\text{HMATSP}}$	5,041	4,900	9,941
kro124	100	$\text{SSY}_{\text{HMATSP}}$	10,000	980,100	990,100
		$\text{SCF}_{\text{HMATSP}}$	10,000	9,801	19,801
rgb443	443	$\text{SSY}_{\text{HMATSP}}$	196,249	86,546,252	86,742,501
		$\text{SCF}_{\text{HMATSP}}$	196,249	195,364	391,613

Results on the comparison of Benders algorithms for the HMATSP

Table 5.6 presents results on the solution of $\text{SCF}_{\text{HMATSP}}$ and $\text{SSY}_{\text{HMATSP}}$ by the standard Benders algorithms denoted by B_{SCF} and B_{SSY}, respectively, on 18 instances derived from the TSPLIB (1997) having 41-443 cities. Columns are identical to those described in Table 5.4. B_{SCF} solved all instances to optimality, while B_{SSY} solved only 12 instances (having up to 100 cities), and it failed to generate a feasible solution to any of the 6 cases having 171-403 cities. Therefore, we can conclude that the Benders-based approach for $\text{SCF}_{\text{HMATSP}}$ is more effective than the Benders-based approach for $\text{SSY}_{\text{HMATSP}}$. Given these results, we only consider B_{SCF} for further investigation.

An important consideration in a Benders algorithm is the generation of strong or nondominated Benders cuts to append to the master problem at each iteration (e.g., see Magnanti and Wong (1981), Papadakos (2008), Fischetti et al. (2010), and Sherali and Lunday (2013)). We have implemented the strategies suggested in Fischetti et al. (2010) for feasibility cuts and in Sherali and Lunday (2013) for both feasibility and optimality cuts. We use the *std*2 method of Fischetti et al. (2010), and for Sherali and Lunday (2013), we use a suitable preemptive perturbation of the right-hand-side of the Benders subproblem (BSP) in order to generate maximal nondominated Benders cuts. The results are presented in Table 5.7 on 16 instances derived from Roberti and Toth (2012), TSPLIB (1997), and Cirasella et al. (2001). B_{S}, B_{FSZ}, and B_{SL} denote, respectively, the standard Benders, the Benders with the cut selection proposed by Fischetti et al. (2010), and the

Table 5.6: Results obtained by solving HMATSP formulations (SCF_{HMATSP} and SSY_{HMATSP}) by standard Benders approaches (B_{SCF} and B_{SSY}) on 18 instances derived from the TSPLIB (1997) having 43-403 cities.

			B_{SCF}		B_{SSY}	
Problem	n	n_i	Z_{IP}	T/gap	Z_{IP}	T/gap
p43	43	U[1,5]	5620	14.96	5620	**11.24**
		U[6,10]	5620	**6.36**	5620	11.22
ftv44	45	U[1,5]	5194	**0.03**	5194	2.43
		U[6,10]	12506	**0.06**	12506	2.65
ftv47	48	U[1,5]	5764	**0.**6	5764	5.18
		U[6,10]	13608	**0.08**	13608	0.16
ry48	48	U[1,5]	41664	**2.12**	41664	5.27
		U[6,10]	102829	**0.37**	102829	4.73
ftv70	71	U[1,5]	6275	**0.23**	6275	19.25
		U[6,10]	15602	**0.34**	15602	26.44
kro124	100	U[1,5]	103840	**18.91**	103840	546.71
		U[6,10]	277397	**1.95**	277397	503.80
ftv170	171	U[1,5]	9422	**131.07**	-	tl
		U[6,10]	22151	**24.37**	-	tl
rgb323	323	U[1,5]	894	**112.79**	-	tl
		U[6,10]	788	**189.45**	-	tl
rgb403	403	U[1,5]	845	**547.25**	-	tl
		U[6,10]	531	**598.41**	-	tl

Table 5.7: Results obtained by solving the proposed single-commodity flow-based formulation for the HMATSP ($\text{SCF}_{\text{HMATSP}}$) by the standard Benders (B_{S}), Benders with the cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and the Benders with nondominated cuts proposed in Sherali and Lunday (2013) (B_{SL}) on 16 instances derived from Roberti and Toth (2012), TSPLIB (1997), and Cirasella et al. (2001) having 43-600 cities.

			B_S		B_{FSZ}		B_{SL}	
Problem	n	n_i	# cuts	T/gap	# cuts	T/gap	# cuts	T/gap
p43	43	U[1,5]	547	14.96	146	666.59	547	**14.35**
		U[6,10]	316	**6.36**	138	149.56	417	13.40
kro100	100	U[1,5]	55	18.91	42	27.58	55	**18.78**
		U[6,10]	3	1.95	4	3.67	3	**1.89**
Balas108	45	U[1,5]	2	**1.03**	3	3.46	2	1.28
		U[6,10]	3	**1.12**	4	4.37	3	1.2
dc112	112	U[1,5]	2986	40.09%	724	**2608.96**	3239	39.92%
		U[6,10]	3027	35.72%	1655	**0.53%**	3527	8.33%
dc134	134	U[1,5]	2551	36.69%	1920	**1.18%**	2455	35.67%
		U[6,10]	2733	43.01%	2009	**0.61%**	2814	41.59%
dc188	188	U[1,5]	1391	42.07%	701	**2.07%**	1327	42.07%
		U[6,10]	898	42.11%	937	**9.63%**	1301	42.01%
dc563	563	U[1,5]	12	tl	7	tl	10	tl
		U[6,10]	5	tl	7	tl	5	tl
atex8	600	U[1,5]	4	tl	3	tl	4	tl
		U[6,10]	2	tl	2	tl	2	tl

Benders with nondominated cuts proposed in Sherali and Lunday (2013). The first three columns are identical to those described in the previous tables. Columns 4-5 display the number of Benders cuts generated and the T/gap obtained by B_{S}. Columns 5-6 and 7-8 (identical to columns 4-5) present the results for B_{FSZ} and B_{SL}, respectively. B_{FSZ} outperformed the others in instances over 100 cities (dc112, dc134, and dc138, and dc188) in terms of the number of cuts generated and minimum T/gap. Therefore, we use only this approach for comparison with the proposed accelerated Benders algorithms presented in the sequel. Note that B_{S}, B_{FSZ}, and B_{SL} were unable to obtain even a feasible solution to instances with 563 and 600 cities.

Table 5.8 presents results obtained by B_{FSZ}, and the accelerated Benders algorithms (A_1-A_4) on 36 instances derived from the TSPLIB (1997) involving 43-443 cities. Columns are identical to those presented in previous tables. B_{FSZ} achieved optimal solutions to all problems except for one instance derived from rbg443, while algorithms A_1-A_4 solved all instances to optimality. For

instances having over 171 cities (rgb323, rgb358, rgb403, and rgb443), the CPU times taken for B_{FSZ} were significantly higher when compared with those reported for A_1-A_4. The average CPU time required for these instances (except for rgb443) by B_{FSZ} and A_4 were 459.65 and 3.24 seconds, respectively. As for comparing algorithms A_1-A_4, we can conclude that A_2 and A_4 outperformed the other approaches by each achieving minimum T/gap in 13 out of 36 cases.

Table 5.9 presents results obtained by the Benders method with the cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and the accelerated Benders algorithms (A_1-A_4) on 36 instances derived from Roberti and Toth (2012) and Cirasella et al. (2001) having 101-1001 cities. Columns are identical to those presented in previous tables. B_{FSZ} obtained optimal solutions for 12 out of the 36 instances, and was unable to generate a feasible solution to any of the 12 instances containing between 563-1001 cities. For the accelerated Benders algorithms, $A_1 - A_4$, each of them solved to optimality 30 out of 36 instances, but A_2 outperformed the others by achieving minimum T/gap in 23 out of the 36 instances. In those 6 instances for which A_2 was unable to obtain optimal solutions, the optimality gaps were less than or equal to 2.25% in 2 out of 6 cases (instances derived from atex8), and less than or equal to 0.16% for the remaining (instances derived from dc832 and dc932).

Given the results presented above, we can infer the following: 1. The accelerated Benders algorithms ($A_1 - A_4$) were the only existing methods that could solve HMATSP instances having over 443 cities; thus outperforming the direct solution of the existing and the proposed single-commodity formulations for the HMATSP by CPLEX as well as the Benders algorithms based on these formulation with and without the generation of strong or nondominated cuts. 2. Generating multiple cuts of the form of (5.4) (Algorithm A_2) each time the subproblem is solved was more effective than generating a unique cut of the form of (5.4) or (5.6) (Algorithms A_1 and A_3, respectively) or multiple cuts of the form of (5.6) (Algorithm A_4). 3. The effectiveness of the accelerated Benders decomposition scheme relies on exploiting of structure of the primal problem, thereby generating feasibility cuts efficiently without having to use a standard LP solver, which might be very time consuming step. 4. We have solved the large-sized HMATSP instances to optimality in the literature.

Table 5.8: Results obtained by solving the proposed single-commodity flow-based formulation for the HMATSP (SCF_{HMATSP}) by the Benders approach with cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and by the proposed accelerated Benders approaches (A_1-A_4) on 36 instances derived from TSPLIB (1997) having 43-443 cities.

				B_{FSZ}		A_1		A_2		A_3		A_4	
Problem	n	n_i	Z_{IP}	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*
p43	43	U[1,5]	5620	134	283.64	197	3.59	59	**0.69**	47	0.72	47	0.83
		U[6,10]	5620	156	140.53	108	3.54	48	**0.66**	46	0.97	46	0.87
ftv44	45	U[1,5]	5194	1	0.06	1	0.03	1	0.03	1	**0.01**	1	0.02
		U[6,10]	12506	1	0.06	1	0.03	1	0.02	1	0.02	1	**0.01**
ftv47	48	U[1,5]	5764	3	0.2	4	0.08	3	0.05	2	0.06	2	**0.04**
		U[6,10]	13608	2	0.17	2	0.05	2	**0.02**	2	0.05	2	0.05
ry48	48	U[1,5]	41644	26	1.72	13	0.23	17	**0.17**	12	0.27	12	0.23
		U[6,10]	102829	17	0.73	11	0.11	12	**0.06**	13	1.00	13	0.17
ft53	53	U[1,5]	21077	9	0.41	6	0.06	7	**0.06**	6	0.11	6	0.09
		U[6,10]	51652	10	0.48	4	0.08	3	**0.03**	3	0.06	3	0.06
ftv55	56	U[1,5]	5142	4	0.48	2	**0.03**	2	0.05	2	0.05	2	0.05
		U[6,10]	11937	8	0.66	3	**0.06**	3	0.06	3	0.08	3	0.09
ftv64	65	U[1,5]	5641	1	0.19	1	0.06	1	0.07	1	0.05	1	**0.02**
		U[6,10]	14770	1	0.25	1	0.08	1	**0.03**	1	0.06	1	0.05
ft70	70	U[1,5]	119659	1	0.33	1	0.06	1	0.11	1	0.07	1	**0.05**
		U[6,10]	308140	3	0.50	2	0.09	2	0.06	1	0.06	1	**0.05**
ftv70	71	U[1,5]	6275	2	0.53	1	0.06	1	0.09	1	0.07	1	**0.04**
		U[6,10]	15602	4	0.86	2	0.14	2	**0.06**	2	0.11	2	0.12
kro124	100	U[1,5]	103840	47	101.28	30	1.86	16	0.27	10	0.66	10	**0.48**
		U[6,10]	277397	4	8.17	2	0.17	2	**0.17**	3	0.22	2	0.2
ftv140	141	U[1,5]	7831	2	20.64	1	0.44	1	0.41	1	0.51	1	**0.39**
		U[6,10]	19329	1	6.33	1	0.23	1	0.23	1	0.27	1	**0.22**
ftv150	151	U[1,5]	8625	3	58.63	3	0.95	3	**0.45**	2	0.69	2	0.62
		U[6,10]	21627	1	216.27	1	0.27	1	0.28	1	0.28	1	**0.25**
ftv160	161	U[1,5]	9331	2	24.27	1	**0.59**	1	0.66	1	0.78	1	0.62
		U[6,10]	21662	1	11.17	1	0.41	1	0.36	1	0.42	1	**0.31**
ftv170	171	U[1,5]	9422	3	72.46	3	1.45	3	**1.09**	2	1.11	2	1.12
		U[6,10]	22151	1	24.27	1	0.72	1	0.66	1	0.53	1	**0.41**
rgb323	323	U[1,5]	894	2	1322.61	1	**3.51**	1	4.45	1	3.53	1	3.76
		U[6,10]	788	1	113.26	1	2.34	1	2.17	1	**2.04**	1	2.28
rgb358	358	U[1,5]	690	1	172.41	1	2.53	1	2.38	1	**2.09**	1	2.31
		U[6,10]	533	2	1152.7	1	**5.30**	1	5.55	1	5.43	1	5.57
rgb403	403	U[1,5]	845	1	455.20	1	4.52	1	**3.73**	1	4.65	1	4.26
		U[6,10]	531	1	299.66	1	**4.2**	1	4.65	1	3.95	1	4.48
rgb443	443	U[1,5]	956	2	tl	1	8.99	1	9.39	1	**8.81**	1	9.24
		U[6,10]	570	1	524.58	1	5.03	1	5.01	1	5.27	1	**4.99**

Table 5.9: Results obtained by solving the single-commodity flow-based formulation (SCF_{HMATSP}) for the HMATSP by the Benders approach with cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and by the proposed accelerated Benders approaches (A_1-A_4) on 36 instances derived from Roberti and Toth (2012) and Cirasella et al. (2001) having 101-1001 cities.

				B_{FSZ}		A_1		A_2		A_3		A_4	
Problem	n	n_i	Z_{IP}	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*
td100.1	101	U[1,5]	31916	1	1.89	1	0.16	1	0.17	1	**0.14**	1	0.17
		U[6,10]	81709	1	1.78	1	0.19	1	0.14	1	0.21	1	**0.11**
Balas108	108	U[1,5]	472	3	8.00	3	**0.20**	3	0.25	4	0.42	4	0.48
		U[6,10]	1046	4	10.45	4	0.23	4	0.25	2	0.22	2	**0.20**
dc112	112	U[1,5]	11109	694	2863.98	234	**10.14**	280	14.96	125	15.04	125	14.94
		U[6,10]	11109	965	8.53%	257	9.47	180	**8.33**	145	12.71	145	13.12
Balas120	120	U[1,5]	916	16	6.48	13	0.70	9	**0.48**	8	1.08	8	1.11
		U[6,10]	2062	16	65.63	9	0.56	10	**0.53**	7	0.85	7	0.83
dc134	134	U[1,5]	5612	458	0.80%	185	9.31	249	**7.5**	125	12.5	125	13.13
		U[6,10]	5612	523	2.30%	346	17.01	264	**10.06**	87	11.7	87	11.97
Balas160	160	U[1,5]	1400	1	11.54	1	0.33	1	0.27	1	**0.25**	1	0.32
		U[6,10]	2917	1	19.78	1	**0.24**	1	0.31	1	0.34	1	0.29
dc176	176	U[1,5]	8587	141	2.94%	420	37.49	389	**26.44**	225	44.87	225	47.10
		U[6,10]	8587	151	54.47%	450	42.85	409	**27.88**	165	32.09	165	33.81
dc188	188	U[1,5]	10225	86	49.96%	917	218.59	542	**62.07**	167	67.83	167	69.76
		U[6,10]	10225	93	55.06%	765	116.39	222	**17.52**	167	49.25	167	49.51
Code198	198	U[1,5]	45058	0	39.59	0	0.61	0	0.57	0	**0.56**	0	0.59
		U[6,10]	9E+09	1	23.60	1	**1.12**	1	1.27	1	1.15	1	1.27
Balas200	200	U[1,5]	1320	22	915.24	12	2.45	13	**1.62**	13	7.41	13	7.38
		U[6,10]	2942	68	2568.5	21	2.89	19	**1.48**	16	6.30	16	6.47
code253	253	U[1,5]	4E+09	1	33.93	1	1.20	1	1.23	1	1.22	1	**1.17**
		U[6,10]	8E+09	1	45.76	1	1.29	1	**1.24**	1	1.27	1	1.25
td316	317	U[1,5]	2279351	1	85.04	1	2.25	1	2.96	1	2.34	1	**2.23**
		U[6,10]	5701444	1	125.47	1	2.40	1	2.37	1	2.98	1	**2.34**
dc563	563	U[1,5]	25951	6	tl	1668	0.38%	1014	**1151.54**	352	1593.97	352	1678.59
		U[6,10]	25591	4	tl	1546	0.38%	1257	**1942.84**	515	2407.92	515	2357.33
atex8	600	U[1,5]	40092	2	tl	1232	91.07%	784	**1.15%**	730	95.32%	732	95.32%
		U[6,10]	40521	6	tl	1216	91.24%	891	**2.25%**	678	95.52%	627	95.49%
big702	702	U[1,5]	79081	2	tl	93	356.43	94	**218.59**	32	295.54	32	298.20
		U(6,10)	79081	2	tl	55	340.25	60	**199.7**	26	395.52	26	393.34
dc895	895	U[1,5]	107775	1	tl	669	53.57%	1573	**0.10%**	352	51.83%	343	51.83%
		U[6,10]	107742	2	tl	665	55.62%	1553	**0.11%**	434	56.76%	415	54.39%
dc932	932	U[1,5]	480207	2	tl	606	tl	1623	**0.09%**	238	tl	239	tl
		U[6,10]	480936	1	tl	1245	tl	1331	**0.16%**	237	tl	300	tl
td1000	1001	U[1,5]	4669875	0	tl	0	64.31	0	65.12	0	65.24	0	**64.28**
		U[6,10]	1.09E+07	0	tl	0	64.91	0	**64.71**	0	64.98	0	64.78

5.5.2 Single-base multiple salesmen HMATSP results

Table 5.10 presents results obtained for solving SB-HMmATSP (SCF_{SB}) directly by CPLEX, by the Benders decomposition with the cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and by the accelerated Benders algorithms ($A_5 - A_8$) on 28 instances derived from the TSPLIB (1997) involving 43-443 cities. The number of salesmen (m) varies between 2 to 4. Column 4 presents the number of salesmen located at the single base, which is denoted by m (the remaining columns are the same as in the previous tables). SCF_{SB} and B_{FSZ} solved to optimality 24 and 28 out of the 28 cases, respectively, but the latter outperformed the former by requiring minimum T/gap in all instances. As for comparing B_{FSZ} and Algorithms $A_5 - A_8$, the latter solved all instances to optimality and outperformed B_{FSZ} by requiring minimum T/gap in all instances. A_6 and A_8, which generate multiple cuts each time the subproblem is solved, outperformed A_5 and A_7 by requiring minimum T/gap for larger-sized instances.

Table 5.11 displays results obtained for solving SB-HMmATSP (SCF_{SB}) directly by CPLEX, by using Benders decomposition with the cut selection proposed by Fischetti et al. (2010) (B_{FSZ}) and the most effective accelerated Benders decomposition schemes (A_6 and A_8) on 36 instances derived from Roberti and Toth (2012) and Cirasella et al. (2001) having 101-1001 cities. The columns are identical to those displayed in the previous table. SCF_{SB} and B_{FSZ} solved to optimality 4 and 6 out of 36 cases, respectively. The proposed algorithms A_6 and A_8 outperformed both SCF_{SB} and B_{FSZ} by solving 24 out of the 36 cases. Furthermore, A_6 outperformed A_8 by requiring minimum T/gap for large-sized instances. For those 12 instances for which A_6 was unable to reach optimality, the optimality gaps were less than or equal to 3.29% for 4 out of 12 instances (instances derived from atex8), and less than or equal to 0.59% for 8 out of 12 cases (instances derived from dc895 and dc932).

We can conclude the following based on the previous results: 1. The accelerated Benders approaches outperformed both the direct solution of the single commodity formulation (SB-HMmATSP) by CPLEX and the Benders decomposition applied to the formulation based on the generation of strong cuts (B_{FSZ}). 2. Generating multiple cuts of the form of (5.4) (algorithm A_6) was more effective than generating a unique cut of the form of 5.4, or (5.41) (algorithms A_5, and A_8, respectively)

or multiple cuts of the form of (5.41) (algorithm A_7). 3. The SB-HMmATSP is a new problem and there does no exist another formulation or solution approach to compare the performance of the proposed algorithms with; however, the effectiveness of the proposed approach is highlighted by the fact that instances with up to 1001 cities could be solved to optimality within one hour of CPU time.

5.5.3 Non-fixed destination multiple base and multiple salesmen HMATSP results

Table 5.12 presents results obtained for solving MBNF-HMmATSP(SCF_{MBNF}) directly by CPLEX , by the Benders decomposition with the cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and by the accelerated Benders algorithms ($A_9 - A_{12}$) on 28 instances derived from the TSPLIB (1997) having 43-443 cities. The number of bases and salesmen vary between 2 to 3 and 3 and 5, respectively. Columns 4-6 present the number of salesmen located at each base, denoted by b_1, b_2, and b_3 (the remaining columns are the same as in the previous tables). B_{FSZ} outperformed SCF_{MBNB} by solving all instances to optimality, while the latter only solved 8 out of the 28 cases. Algorithms $A_9 - A_{12}$ outperformed B_{FSZ} by requiring minimum T/gap in all instances. A_{10} and A_{12} dominated A_9 and A_{11} by requiring minimum T/gap for the larger-sized instances.

Table 5.13 presents results obtained for solving MBNF-HMmATSP (SCF_{MBNF}) directly by CPLEX, by the Benders decomposition with the cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and the most effective accelerated Benders algorithms (A_{10} and A_{12}) on 36 instances derived from Roberti and Toth (2012) and Cirasella et al. (2001) having 101-1001 cities. Columns are identical to those presented in previous table. SCF_{MBNF} and B_{FSZ} solved to optimality 3 and 8 out of the 36 instances, respectively. A_{10} outperformed all other approaches by requiring minimum T/gap for large-sized instances and by solving to optimality 25 out of the 36 cases. For those 11 instances for which A_{10} was unable to reach optimality, the optimality gaps were less than or equal to 2.64% in 3 out of 11 instances (instances derived from atex8), and less than or equal to 0.16% in 8 out of 11 cases (instances derived from dc895 and dc932).

Given the results presented above, we can infer the following: 1. The accelerated Benders approaches outperform both the direct solution of the single commodity formulation (MBNF-HMmATSP) by CPLEX and the Benders-based method applied to the formulation along with the generation of strong cuts (B_{FSZ}). 2. Generating multiple cuts of the form of (5.57) (Algorithm A_{10}) is more effective than generating a unique cut of the form (5.57), or (5.58) (Algorithms A_9 and A_{10}, respectively) or multiple cuts of the form of (5.58) (Algorithm A_{12}). 3. The MBNF-HMmATSP is a new problem, and there does not exist methods to compare the performance of the proposed algorithms with; however, their effectiveness is highlighted by the fact that instances with up to 1001 cities can be solved to optimality within one hour of CPU time.

Table 5.10: Results obtained by solving SB-HMmATSP using the single-commodity flow-based formulation (SCF_{SB}), the Benders with cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and by the proposed accelerated Benders approaches (A_5-A_8) on 28 instances derived from TSPLIB (1997) having 43-443 cities.

					SCF_{SB}		B_{FSZ}		A_5		A_6		A_7		A_8	
Problem	n	n_i	m	Z_{IP}	# cuts	T/gap	# cuts	T/gap	# cuts	T/gap	# cuts	T/gap	# cuts	T/gap	# cuts	T/gap
p43	53	U(1,5)	2	5633	-	0.68%	106	80.76	329	2.65	54	**0.44**	47	1.58	47	1.50
			4	5673	-	0.76%	104	96.74	246	2.43	54	**0.47**	45	0.87	45	0.78
		U(6,10)	2	5633	-	0.68%	162	1334.40	244	2.25	50	0.67	42	0.76	42	**0.66**
			4	5673	-	0.45%	94	63.73	241	2.23	50	**0.45**	37	0.73	37	0.92
kro124	100	U(1,5)	2	0	-	0.00	33	21.75	58	3.87	17	**0.45**	10	0.53	10	0.45
			4	0	-	37.41	31	21.04	57	3.92	14	**0.20**	7	0.58	7	0.39
		U(6,10)	2	0	-	4.73	4	3.26	3	0.41	2	**0.12**	2	0.28	2	0.22
			4	0	-	4.76	4	3.42	3	0.34	2	**0.19**	2	0.19	2	0.27
ftv170	171	U(1,5)	2	0	-	0.00	3	11.61	2	1.28	3	1.14	2	1.19	2	**1.05**
			4	0	-	0.00	2	9.13	1	0.66	1	0.76	1	0.87	1	**0.72**
		U(6,10)	2	0	-	0.00	1	6.72	1	0.50	1	0.46	1	0.49	1	**0.34**
			4	403	-	0.00	1	5.68	1	0.43	1	0.45	1	0.47	1	**0.37**
rgb323	323	U(1,5)	2	908	-	714.33	2	74.32	1	4.04	1	**3.24**	1	3.35	1	3.65
			4	938	-	335.03	2	71.34	1	3.38	1	**3.23**	1	3.34	1	3.65
		U(6,10)	2	800	-	356.43	1	40.26	1	**1.95**	1	2.07	1	2.07	1	2.01
			4	826	-	245.55	3	105.07	2	5.21	1	4.17	1	4.40	1	**3.90**
rgb358	358	U(1,5)	2	697	-	1088.59	2	98.22	2	8.27	2	**4.41**	3	8.30	3	8.63
			4	711	-	1082.5	1	55.52	1	4.10	1	**2.29**	1	2.31	1	2.43
		U(6,10)	2	539	-	902.09	4	208.12	3	10.59	1	**4.85**	1	5.12	1	4.87
			4	553	-	543.79	2	100.96	1	6.00	2	7.14	1	**4.82**	1	5.44
rbg403	403	U(1,5)	2	851	-	811.31	3	189.92	3	12.87	2	**7.04**	2	9.86	2	9.73
			4	868	-	1466.9	2	121.48	1	6.97	1	7.91	1	7.50	1	**7.13**
		U(6,10)	2	530	-	1366.7	1	71.14	1	**3.71**	1	3.73	1	3.98	1	3.71
			4	542	-	1234.6	1	74.38	1	4.21	0	3.74	1	3.87	1	**3.76**
rbg443	443	U(1,5)	2	962	-	2875.60	1	95.64	1	5.15	0	**4.84**	1	4.91	1	5.30
			4	974	-	2452.4	1	89.72	1	5.29	1	**4.67**	1	4.77	1	4.95
		U(6,10)	2	570	-	645.78	0	90.18	0	5.09	0	4.84	0	4.14	0	**3.65**
			4	571	-	697.46	0	89.31	0	4.93	0	**3.92**	0	5.26	0	4.87

Table 5.11: Results obtained by solving SB-HMmATSP using the single-commodity flow-based formulation (SCF_{SB}), the Benders with cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and by the proposed accelerated Benders approaches (A_6, and A_8) on 36 instances derived from Roberti and Toth (2012) and Cirasella et al. (2001) having 101-1001 cities.

					SCF_{SB}		B_{FSZ}		A_6		A_8	
Problem	n	n_i	m	Z_{IP}	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*
dc112	112	U(1,5)	2	11241	-	0.47%	1332	0.39%	98	**2.42**	85	6.15
			4	11538		0.18%	853	1063.44	94	**2.00**	65	4.88
		U(6,10)	2	11241		0.13%	923	2749.11	87	**2.01**	84	5.73
			4	11538		0.11%	1640	0.45%	108	**2.62**	110	6.74
dc188	188	U(1,5)	2	10425	-	17.49%	86	48.93%	623	53.90	126	**37.32**
			4	10843	-	2.90%	117	1.26%	320	**22.54**	145	38.95
		U(6,10)	2	10425	-	0.91%	114	53.85%	143	44.38	154	**37.44**
			4	10843	-	1.35%	113	52.61%	180	**168.31**	173	212.2
Balas200	200	U(1,5)	2	1320	-	243.30	20	119.11	14	**1.89**	13	6.47
			4	1323	-	1037.28	24	147.95	12	**1.23**	13	5.83
		U(6,10)	2	2937	-	381.42	60	320.97	16	**1.54**	17	7.29
			4	2929	-	455.15	85	392.92	22	**2.31**	20	8.35
dc563	563	U(1,5)	2	26042	-	tl	7	tl	1356	**1492.56**	384	1905.27
			4	26227	-	tl	7	tl	881	**1265.29**	387	1671.94
		U(6,10)	2	26042	-	tl	7	tl	1470	**1633.94**	496	2293.03
			4	26227	-	tl	7	tl	1195	**1123.75**	403	1944.99
atex8	600	U(1,5)	2	40118	-	tl	5	tl	902	**1.77%**	519	96.29%
			4	40586	-	tl	4	tl	880	**2.92%**	687	95.58%
		U(6,10)	2	40240	-	tl	5	tl	917	**2.12%**	640	96.29%
			4	40743	-	tl	6	tl	868	**3.29%**	677	95.58%
big702	702	U(1,5)	2	79281	-	tl	4	tl	33	**125.86**	22	302.41
			4	79881	-	tl	11	tl	26	**118.11**	24	325.82
		U(6,10)	2	79281	-	tl	6	tl	62	**194.67**	24	317.88
			4	79881	-	tl	8	tl	56	**171.65**	28	382.75
dc895	895	U(1,5)	2	108156	-	tl	3	tl	1714	**0.09%**	514	53.30%
			4	108906	-	tl	2	tl	1843	**0.05%**	408	tl
		U(6,10)	2	108185	-	tl	1	tl	1626	**0.13%**	425	56.73%
			4	108920	-	tl	1	tl	1865	0.11%	370	56.59%
dc932	932	U(1,5)	2	480084	-	tl	2	tl	1695	**0.05%**	287	tl
			4	480267	-	tl	1	tl	1671	**0.59%**	304	tl
		U(6,10)	2	480372	-	tl	3	tl	1579	**0.12%**	296	2.61%
			4	480116	-	tl	2	tl	1300	**0.05%**	333	tl
td1000	1001	U(1,5)	2	4668874	-	tl	0	tl	0	63.1	0	**63.02**
			4	4668585	-	tl	0	tl	0	65.36	0	**61.76**
		U(6,10)	2	1.09E+07	-	tl	0	tl	0	71.35	0	**70.72**
			4	1.09E+07		tl	0	tl	0	**66.19**	0	69.87

Table 5.12: Results obtained by solving NFMB-HMmATSP using the single-commodity flow-based formulation (SCF_{MBNF}), the Benders with cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and by the proposed accelerated Benders approaches (A_8-A_{12}) on instances from TSPLIB (1997) having 43-443 cities.

							SCF_{MBNF}		B_{FDZ}		A_9		A_{10}		A_{11}		A_{12}	
Problem	n	n_i	b_1	b_2	$b3$	Z_{IP}	# cuts	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*
p43	43	U(1,5)	1	2	-	5582	-	1.37%	141	331.74	56	0.73	56	**0.45**	30	0.70	30	0.72
			1	1	2	5582	-	50.54%	101	68.3	47	0.36	44	**0.22**	47	1.56	47	1.48
		U(6,10)	1	2	-	5582	-	0.53%	121	528.25	47	**0.25**	42	0.30	39	1.36	39	1.15
			1	1	2	5582	-	0.82%	112	119.93	43	0.31	37	**0.28**	33	0.56	33	0.56
kro124	100	U(1,5)	1	2	-	103325	-	24.12	54	2511.21	13	0.39	13	**0.17**	9	0.34	9	0.33
			1	1	2	103099	-	27.02	31	601.03	11	0.27	12	**0.20**	8	0.39	8	0.33
		U(6,10)	1	2	-	273381	-	4.85	9	472.15	3	0.11	4	0.12	3	**0.2**	3	0.16
			1	1	2	271413	-	3.68	8	278.20	4	**0.09**	4	0.11	3	0.17	3	0.17
ftv170	171	U(1,5)	1	2	-	9417	-	203.43	2	8.97	1	0.71	1	**0.62**	1	0.64	1	0.67
			1	1	2	9400	-	234.39	2	8.27	1	0.75	1	0.67	1	0.72	1	**0.59**
		U(6,10)	1	2	-	22099	-	23.95	1	6.10	0	0.49	0	0.41	0	0.42	0	**0.38**
			1	1	2	22099	-	28.58	1	6.75	0	0.46	0	0.37	0	**0.35**	0	0.37
rgb323	323	U(1,5)	1	2	-	901	-	tl	3	109.59	2	3.18	3	**3.06**	2	5.09	2	5.13
			1	1	2	903	-	tl	3	103.27	2	3.04	2	**3.03**	2	4.48	2	4.6
		U(6,10)	1	2	-	794	-	tl	1	40.65	1	2.14	1	**1.73**	1	1.97	1	1.82
			1	1	2	797	-	tl	3	110.71	2	7.97	2	**3.34**	2	4.73	2	4.74
rgb358	358	U(1,5)	1	2	-	697	-	tl	2	105.59	1	**4.06**	1	4.20	1	4.06	1	4.07
			1	1	2	697	-	tl	3	154.75	2	4.26	2	**4.24**	2	5.9	2	5.99
		U(6,10)	1	2	-	539	-	tl	2	110.15	1	**4.20**	1	5.09	1	4.89	1	4.73
			1	1	2	539	-	tl	2	109.22	1	5.82	1	4.61	1	**4.15**	1	4.87
rbg403	403	U(1,5)	1	2	-	852	-	tl	1	79.7	1	3.84	1	**3.81**	1	3.92	1	4.10
			1	1	2	845	-	tl	0	73.96	0	**3.37**	0	3.42	0	3.91	0	3.54
		U(6,10)	1	2	-	535	-	tl	0	74.26	0	3.51	0	3.39	0	**3.34**	0	3.37
			1	1	2	551	-	tl	0	76.71	0	**3.51**	0	3.65	0	3.54	0	3.56
rbg443	443	U(1,5)	1	2	-	957	-	tl	1	89.51	1	4.15	1	**4.13**	1	4.27	1	4.48
			1	1	2	957	-	tl	2	186.19	2	4.06	2	4.07	2	3.95	2	**3.84**
		U(6,10)	1	2	-	570	-	tl	1	88.16	1	**4.54**	1	4.56	1	4.65	1	5.38
			1	1	2	573	-	tl	1	89.11	1	4.49	1	**4.37**	1	4.51	1	4.40

Table 5.13: Results obtained by solving NFMB-HMmATSP using the single-commodity flow-based formulation (SCF_{MBNF}), the Benders with cut selection proposed by Fischetti et al. (2010) (B_{FSZ}), and by the proposed accelerated Benders approaches (A_{10} and A_{12}) on instances derived from Roberti and Toth (2012) and Cirasella et al. (2001) having 101-1001 cities.

							SCF_{MBNF}		B_{FDZ}		A_{10}		A_{12}	
Problem	n	n_i	b_1	b_2	b_3	Z_{IP}	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*	*# cuts*	*T/gap*
dc112	112	U(1,5)	1	2	-	11304	-	0.86%	576	727.67	163	**3.81**	88	5.04
			1	1	2	11325	-	1.85%	820	1278.01	114	**2.00**	82	5.16
		U(6,10)	1	2	-	11304	-	4.29%	541	483.63	111	**2.14**	97	6.1
			1	1	2	11325	-	26.03%	1075	0.03%	113	**2.06**	79	6.85
dc188	188	U(1,5)	1	2	-	10444	-	tl	623	1.26%	238	**13.79**	139	38.03
			1	1	2	10449	-	39.90%	723	1.30%	510	**53.74**	143	39.03
		U(6,10)	1	2	-	10444	-	67.46%	526	476.28	471	47.17	125	**37.66**
			1	1	2	10449	-	tl	862	2.41%	679	77.05	122	**32.87**
Balas200	200	U(1,5)	1	2	-	1316		1099.62	21	127.39	16	**1.81**	14	5.12
			1	1	2	1321	-	0.09%	16	103.58	15	**1.06**	12	5.3
		U(6,10)	1	2	-	2947	-	345.23	460	477.29	18	**1.36**	16	6.47
			1	1	2	2986	-	402.72	64	342.73	13	**3.56**	15	6.72
dc563	563	U(1,5)	1	2	-	26058	-	tl	6	tl	1144	**1038.78**	383	1598.68
			1	1	2	26083	-	tl	8	51.83%	1008	**1414.70**	458	2015.31
		U(6,10)	1	2	-	26058	-	tl	7	tl	1093	**1160.4**	490	1950.96
			1	1	2	26082	-	tl	7	tl	1323	**1386.4**	347	1490.12
atex8	600	U(1,5)	1	2	-	40563	-	tl	2	tl	893	**2.64%**	669	95.52%
			1	1	2	39521	-	tl	3	tl	1233	**1343.81**	652	96.30%
		U(6,10)	1	2	-	40539	-	tl	4	tl	900	**2.27%**	694	94.45%
			1	1	2	39654	-	tl	3	tl	776	**1.37%**	688	96.26%
big702	702	U(1,5)	1	2	-	79281	-	tl	2	tl	82	**239.45**	22	257.35
			1	1	2	79481	-	tl	4	tl	68	**123.51**	46	448.03
		U(6,10)	1	2	-	79281	-	tl	11	tl	64	**124.64**	43	500.50
			1	1	2	79481	-	tl	6	tl	58	**189.21**	28	340.72
dc895	895	U(1,5)	1	2	-	108429	-	tl	5	tl	1646	**0.15%**	393	53.39%
			1	1	2	108368	-	tl	4	tl	1471	**0.11%**	370	52.08%
		U(6,10)	1	2	-	108367	-	tl	3	tl	1777	**0.12%**	369	54.96%
			1	1	2	107736	-	tl	2	tl	1559	**0.07%**	424	56.14%
dc932	932	U(1,5)	1	2	-	479999	-	tl	2	tl	1153	**0.04%**	265	tl
			1	1	2	480265	-	tl	5	tl	1502	**0.16%**	327	tl
		U(6,10)	1	2	-	480341	-	tl	3	tl	1301	**0.12%**	248	tl
			1	1	2	480207	-	tl	4	tl	1329	**0.08%**	295	tl
td1000	1001	U(1,5)	1	2	-	4669875	-	tl	4	tl	0	**57.75**	0	58.39
			1	1	2	4668882	-	tl	0	tl	0	64.04	0	64.98
		U(6,10)	1	2	-	1.08E+07	-	tl	0	tl	0	**67.08**	0	68.91
			1	1	2	1.08E+07	-	tl	0	tl	0	**69.17**	0	69.71

Chapter 6

A New Formulation and Exact Approaches for the Fixed-Destination Multi-depot Traveling Salesman Problem with Load Balancing (FD-MTSPB)

6.1 Introduction

The fixed-destination multi-depot traveling salesman problem with load balancing (FD-MTSPB) can be concisely defined as follows:

Given a complete graph with vertex set $N = D \cup N'$, where the first $|D|$ nodes of N comprise a set of depots having m_d salesmen located initially at depot $d \in D$; and where $N' = \{|D|+1, |D|+2, ..., n\}$ comprises the set of cities to be visited, and given an asymmetric distance matrix $[c_{ij}]$, find $m = \sum_{d \in D} m_d$ tours, with m_d tours starting and ending at depot $d \in D$, while collectively having visited

city i exactly once, $\forall i \in N'$, such that the total distance traveled is minimized and the number of visits made by each salesman lie in the interval [L,K], where L and K are specified positive integers.

Fixed destination routing and logistics problems require vehicles or salesmen located at the beginning of the day at a given point (e.g., home city) to return to that point at the end of the day. Examples include variants of the vehicle routing problems (Golden et al. (1977) and Laporte et al. (1988)), the multiple traveling salesmen problem (Kara and Bektaş (2006) and Burger (2014)), and postal distribution, less-than-truckload transport operations, the traveling repairmen problem, and balance billing-cycle vehicle routing problems (Bektaş (2012)).

There are less than a handful of polynomial-length formulations for the fixed destination multi-depot traveling salesmen problem (FD-MTSP) available in the literature (Kara and Bektaş (2006), Bektaş (2012), and Burger (2014)). Recently, Bektaş (2012) introduced the FD-MTSP with load balancing (FD-MTSPB), and proposed several formulations for this problem along with Benders decomposition-based methodologies for their solution, which outperformed the direct application of CPLEX on problem instances containing up to 171 cities. In this chapter, we present a new flow-based formulation for the FD-MTSPB that is particularly amenable to solution using Benders decomposition.

An interesting extension of the FD-MTSPB, which has not been addressed in the literature, is the high-multiplicity FD-MTSPB (HMFD-MTSPB), where a city must be visited multiple times. The HMFD-MTSPB problem arises during the batch production of different product types on multiple machines, where the machines (salesmen) process product batches (cities) for a specified number of times while requiring sequence-dependent set-up costs, and where each machine is required to process a given number of batches in a prescribed interval and return to its starting position. The high multiplicity traveling salesmen problem (HMTSP) has been studied by Grigoriev and van de Klundert (2006), and Sarin et al. (2011).

The remainder of this chapter is organized as follows. In Section 6.2, we present existing and new formulations for the FD-MTSPB, the non-fixed MTSPB and the high multiplicity FD-MTSPB problems. In Section 6.3, we develop Benders decomposition-based approaches for the formulation presented by Burger (2014) that is adapted to the FD-MTSPB, as well as for our proposed formu-

lation and for the non-fixed MTSPB. In Section 6.4, we introduce a two-phase approach to solve the FD-MTSPB, and illustrate it using a simple example. In Section 6.5, we present computational results for solving the proposed formulations by CPLEX, Benders-based approaches, and the proposed two-phase algorithm using benchmark problem instances available in the literature. We also compare our results with those obtained using the same dataset by Bektaş (2012).

6.2 Mathematical formulations for the FD-MTSPB and related problems

In this section, we present formulations for the FD-MTSPB, the non-fixed destination FD-MTSPB (NF-MTSPB), and the high multiplicity FD-MTSPB (HMFD-MTSPB).

6.2.1 Formulations for the FD-MTSPB and NF-MTSPB

Consider the following notation:

Sets and parameters:

N	:	Set of all cities, with $\|N\| = n$.
D	:	Set of depots, $D = \{1, ..., \|D\|\}$.
N'	:	Set of cities to be visited, $N' = \{\|D\| + 1, \|D\| + 2, ..., \|N\|\}$.
c_{ij}	:	Distance from city i to city j, $i, j \in N$.
m_d	:	Number of salesmen located at depot $d \in D$.
$[L, K]$	:	Range of the number of visits to be made by each salesman, where L and K are positive integers.

Decision variables:

x_{ij}	:	Equals 1 if arc (i, j) is used in the solution, and equals 0 otherwise, $\forall i, j \in N$.
y_{ij}	:	Flow on arc (i, j), $i \neq j$, $\forall i, j \in N$.

z_{ij}^d : Flow on arc (i,j) from depot d, $\forall i,j \in N$, $i \neq j$, $\forall d \in D$.

k_i : Variable that indicates the label assigned to city i, $\forall i \in N$.

y'_{ij} : Variable that indicates the label assigned to arc (i,j), $i \neq j$, $\forall i,j \in N$.

We begin by presenting a formulation for the FD-MTSPB due to Bektaş (2012), except that contrary to (Bektaş, 2012), we do not replicate depots.

FD-MTSPB1 :

$$\text{Minimize} \quad \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \tag{6.1}$$

subject to

$$\sum_{i \in N'} x_{di} = m_d, \quad \forall d \in D \tag{6.2}$$

$$\sum_{i \in N'} x_{id} = m_d, \quad \forall d \in D \tag{6.3}$$

$$\sum_{j \in N} x_{ji} = 1, \quad \forall i \in N' \tag{6.4}$$

$$\sum_{j \in N} x_{ij} = 1, \quad \forall i \in N' \tag{6.5}$$

$$y_{ij} \leq K x_{ij}, \quad \forall i,j \in N \tag{6.6}$$

$$y_{di} \geq L x_{di}, \quad \forall d \in D, i \in N' \tag{6.7}$$

$$\sum_{j \in N,\ j \neq i} y_{ji} - \sum_{j \in N',\ j \neq i} y_{ij} = 1, \quad \forall i \in N' \tag{6.8}$$

$$z_{ij}^d \leq x_{ij}, \quad \forall i,j \in N, \quad d \in D \tag{6.9}$$

$$\sum_{i \in N'} z_{di}^d = m_d, \quad d \in D \tag{6.10}$$

$$\sum_{i \in N'} z_{id}^d = m_d, \quad d \in D \tag{6.11}$$

$$\sum_{j \in N,\ j \neq i} z_{ji}^d - \sum_{j \in N',\ j \neq i} z_{ij}^d = 0, \quad \forall d \in D, \quad \forall i \in N' \tag{6.12}$$

$$x_{ij} \in \{0,1\}, \quad \forall i,j \in N, \quad i \neq j \tag{6.13}$$

$$y_{ij} \geq 0, \quad \forall i,j \in N, \quad i \neq j \tag{6.14}$$

$$z_{ij}^d \geq 0, \quad \forall i,j \in N, \quad i \neq j, \quad \forall d \in D. \tag{6.15}$$

Constraints (6.2) and (6.3), respectively, enforce that m_d salesmen depart and return to depot d, $\forall d \in D$. Constraints (6.4) and (6.5) assure that each city is visited and departed exactly once. Constraints (6.6)-(6.8) are subtour elimination constraints, and they impose the connectivity of the graph induced by the **x**-variables by requiring a path defined by the flow variables y_{ij} to exist from each depot, $d \in D$, to each city i, $i \in N'$, where one unit flows from depots into each city i, and where Constraints (6.6) and (6.7), respectively, impose the load balancing restrictions that each path carries at most K and at least L units of flow. Constraints (6.9)-(6.12) impose the fixed destination feature of the problem by requiring m_d units of flow to leave and come back to each depot d using only those arcs for which $x_{ij} > 0$.

Next, we present another formulation for the FD-MTSPB due to Burger (2014), which is given as follows:

FD-MTSPB2 :

Minimize (6.1)

subject to

(6.2) − (6.8), and (6.13) − (6.14)

$$k_d = d, \quad d \in D \tag{6.16}$$

$$k_i - k_j + (|D| - 1)x_{ij} \leq |D| - 1, \quad \forall i, j \in N, \quad i \neq j \tag{6.17}$$

$$k_i \geq 0, \quad \forall i \in N. \tag{6.18}$$

Constraints (6.16) enforce that for each depot, $d \in D$, $k_d = d$. Constraints (6.17) label the cities visited from each $d \in D$ based on the particular index d so that an exchange of salesmen among depots leads to a contradiction. Constraints (6.18) capture the domain of the decision variables.

Our new model formulation for the problem, on the other hand, is as follows:

FD-MTSPB3 :

Minimize (6.1)

subject to

(6.2) − (6.8), and (6.13) − (6.14)

$$y'_{ij} \leq |D| x_{ij}, \quad \forall i, j \in N \tag{6.19}$$

$$y'_{di} = dx_{di}, \quad \forall d \in D, \quad i \in N' \tag{6.20}$$

$$y'_{id} = dx_{id}, \quad \forall d \in D, \quad i \in N' \tag{6.21}$$

$$\sum_{j \in N,\ j \neq i} y'_{ji} - \sum_{j \in N,\ j \neq i} y'_{ij} = 0, \quad \forall i \in N' \tag{6.22}$$

$$y'_{ij} \geq 0, \quad \forall i, j \in N, \quad i \neq j. \tag{6.23}$$

In this formulation, a label d is assigned to all the salesmen leaving from depot d, and this label is used as a flow that is maintained throughout the tours of the salesmen from that depot. To this end, Constraints (6.19) enforce flow to only occur on arcs for which $x_{ij} > 0$. Constraints (6.20) and (6.21) enforce d units to depart and return to each base using those arcs for which $x_{di} > 0$ and $x_{id} > 0$, $d \in D$, $i \in N'$, respectively. Constraints (6.22) are the standard flow conservation constraints. Constraints (6.23) represent the domain of the y'_{ij}-variables.

Note that x_{ij} and y_{ij} are common decision variables in both FD-MSTPB1 and FD-MTSPB3. Furthermore, Constraints (6.9)-(6.12) in FD-MTSPB1 ensure that m_d units of flow leave and return to the same $d \in D$ using the arcs for which $x_{ij} > 0$. Constraints (6.19)-(6.23) ensure the same in formulation FD-MTSPB3. In fact, if we let $y'_{ij} = \sum_{d \in D} dz^d_{ij}$, $\forall i, j \in N$, then, y'_{ij}, $\forall i, j \in N$, satisfy Constraints (6.19)-(6.23). Similarly, a feasible path for a d in FD-MTSPB3 is equivalent to setting $z^d_{ij} = 1$ $\forall (i, j)$ on this path, and $z^d_{ij} = 0$, otherwise $\forall d \in D$. Consequently, z^d_{ij}, $\forall i, j \in N$, $\forall d \in D$, will satisfy Constraints (6.9)-(6.12). Hence, FD-MTSPB1 and FD-MTSPB3 are equivalent. Note that FD-MTSPB2 labels the cities, while FD-MTSPB3 labels the arcs visited by each $d \in D$.

We add to FD-MTSPB1, FD-MTSPB2, FD-MTSPB3 the following two-city subtour elimination constraint:

$$x_{ij} + x_{ji} \leq 1, \quad \forall i, j \in N', \tag{6.24}$$

and following Bektaş (2012), we replace (6.6) and (6.7) by a strengthened version given by:

$$y_{di} \leq K x_{di}, \quad \forall d \in D, i \in N' \tag{6.25}$$

$$y_{ij} \leq (K - 1) x_{ij}, \quad \forall i, j \in N' \tag{6.26}$$

$$y_{di} \geq L x_{di}, \quad \forall d \in D, i \in N'. \tag{6.27}$$

Let z_{LP}(B1), z_{LP}(B2) and z_{LP}(B3), respectively, denote the objective values of the linear programming relaxations of FD-MTSPB1, FD-MTSPB2 , and FD-MTSPB3. Then, we have the following

result:

Proposition 6.2.1. *There does not exist a dominance relationship between $z_{LP}(B1)$ and $z_{LP}(B2)$, $z_{LP}(B1)$ and $z_{LP}(B3)$, and between $z_{LP}(B2)$ and $z_{LP}(B3)$.*

Proof. We show this by example. Consider Instance #1 in Table 6.2 in Section 6.5.1. Note that z_{LP}(B1) = 1425.17, z_{LP}(B2) = 1422.65, and z_{LP}(B3) = 1424.33. However, for Instance #4, z_{LP}(B1) = 1979.98, z_{LP}(B2) = 2055.86, and z_{LP}(B3) = 2055.86.

To show that there does not exist a dominance relationship between z_{LP}(B2) and z_{LP}(B3), we present an instance described in Figure 6.1. In this example we have, $n = 10$ (cities), $L = 2$, $K = 5$, and cities 1 and 2 are depots (squares) at each of which we have a unique salesman ($m_1 = 1$ and $m_2 = 1$). The rows of the underlying matrix, $[c_{i,j}]$, are as follows:= {5, 6, 1, 9, 9, 9, 8, 9, 7; 9, 3, 1, 7, 5, 4, 8, 1, 7; 2, 5, 7, 6, 4, 6, 3, 6, 5; 9, 2, 1, 4, 9, 3, 9, 6, 7; 4, 8, 9, 8, 4, 8, 5, 10, 2; 10, 8, 2, 5, 7, 5, 6, 9, 8; 2, 4, 2, 6, 3, 9, 7, 5, 9; 9, 2, 10, 8, 4, 10, 4, 1, 10; 9, 10,7, 4, 1, 3, 3, 3, 4; 6, 6, 5, 9, 3, 2, 10, 6, 3}. For this instance, z_{LP}(B2) = 19.33 and z_{LP}(B3) = 19.25, However, for Instance #1 in Table 6.2 in Section 6.5.1, z_{LP}(B2) = 1422.65, and z_{LP}(B3) = 1424.33 □

Finally, we present a formulation for the non-fixed destination multi-depot traveling salesman problem with load balancing (NF-MTSPB), which is used in our two-phase solution approach described in Section 6.4. This is given as follows:

NF-MTSPB: Minimize (6.1); subject to (6.2)-(6.5), (6.8), (6.13)-(6.14), and (6.24)- (6.27).

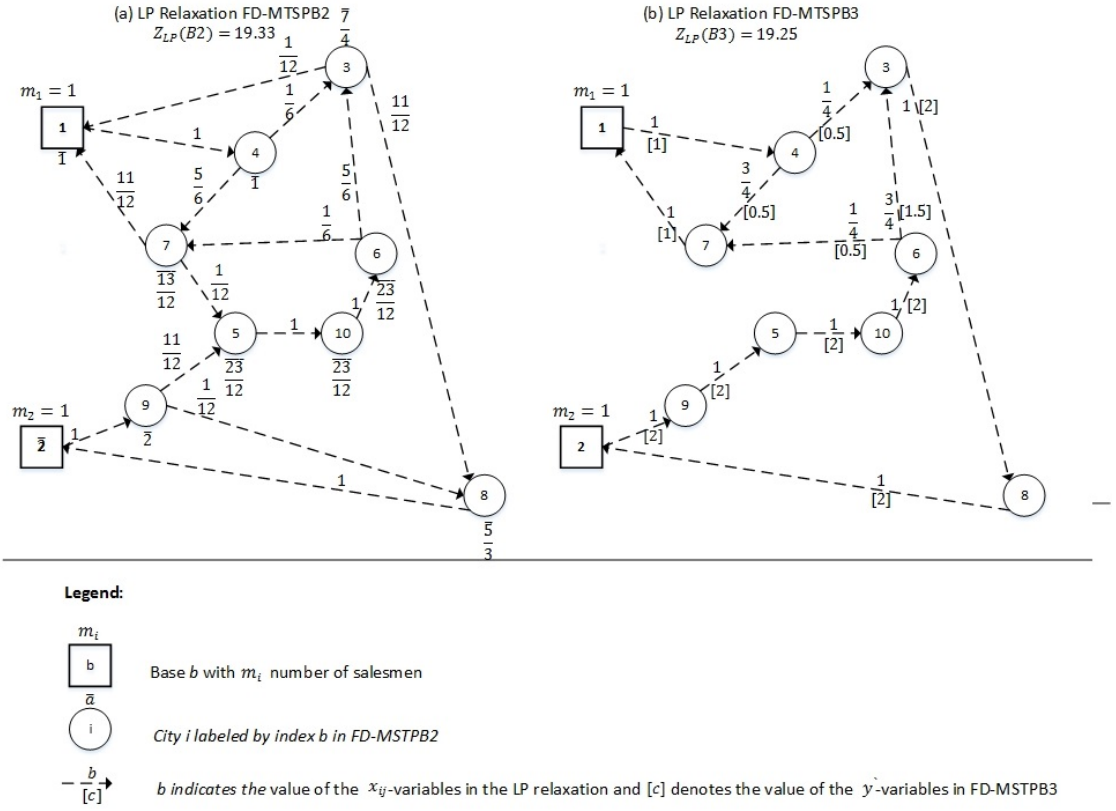

Figure 6.1: Comparison between LP relaxations.

6.2.2 Formulations for the high-multiplicity FD-MTSPB

Let n_i denote the number of visits required for city i, $\forall i \in N'$. Then, the high-multiplicity fixed destination multi-depot traveling salesmen problem with load balancing (HMFD-MTSPB) can be formally defined as follows:

Given a complete graph with vertex set $N = D \cup N'$, where the first $|D|$ nodes of N comprise a set of depots having m_d salesmen located initially at depot $d \in D$, and where $N' = \{|D|+1, |D|+2, ..., n\}$ comprises the set of cities to be visited, and given an asymmetric distance matrix $[c_{ij}]$, and a positive integer number n_i for each $i \in N'$, find $m = \sum_{d \in D} m_d$ tours, with m_d tours starting and ending at depot $d \in D$, while collectively having visited city i exactly n_i times, $\forall i \in N'$, such that the total distance traveled is minimized and the number of visits made by each salesman is between a given interval denoted by [L,K], where L and K are specified positive integers.

Below, we present two formulations for the HMFD-MTSPB. The first one is stated as follows:

HMFD-MTSPB1 :

Minimize (6.1)

subject to

$(6.2) - (6.3)$, $(6.9) - (6.12)$, $(6.14) - (6.15)$, and $(6.25) - (6.27)$

$$\sum_{j \in N} x_{ji} = n_i, \quad \forall i \in N' \tag{6.28}$$

$$\sum_{j \in N} x_{ij} = n_i, \quad \forall i \in N' \tag{6.29}$$

$$\sum_{j \in N,\ j \neq i} y_{ji} - \sum_{j \in N - \{1\},\ j \neq i} y_{ij} = n_i, \quad \forall i \in N' \tag{6.30}$$

$$x_{ij} \geq 0 \text{ and integer}, \quad \forall i, j \in N. \tag{6.31}$$

First of all, note that in HMFD-MTSPB1, x_{ij} is restricted to be integer-valued. Constraints (6.2)-(6.3) ensure that m_d salesmen depart and return to depot $d \in D$. As before, Constraints (6.9)-(6.12) impose the fixed destination feature by requiring m_d units of flow to leave and return to depot $d \in D$ using only those arcs for which $x_{ij} > 0$. Constraints (6.14)-(6.15) represent the domains of the y_{ij}- and z_{ij}-decision variables. Constraints (6.25)-(6.27), respectively, impose the load balancing restrictions that each path carries at most K and at least L units of flow. The

range $[L, K]$ was modified accordingly to generate feasible solutions when x_{ij} is restricted to be integer-value. Constraints (6.28) and (6.29) enforce that each city i is visited and departed exactly n_i times. Constraints (6.30) play the role of (6.8), but now require n_i units to flow from the depots into each $i \in N'$ while balancing loads. Constraints (6.31) represent the logical requirements (note that in this formulation self-loops are permitted, i.e, x_{ii} can take an integer value, $i \in N'$).

The second formulation is stated as follows:

HMFD-MTSPB2: Minimize (6.1); subject to (6.2)-(6.3), (6.14), (6.19)-(6.23), (6.25)-(6.27), and (6.28)-(6.31).

Constraints (6.2)-(6.3), (6.14), (6.25)-(6.27), and (6.28)-(6.31) are as defined before. When x_{ij} is integer-valued, Constraints (6.19)-(6.23) impose a label dx_{di} instead of d (since $x_{di} \in \{0, 1\}$ in formulation FD-MTSPB3) to be maintained throughout the tours of the salesmen from depot $d \in D$.

Note that if $n_i = 1, \forall i \in N'$, then HMFD-MTSPB2 reduces to FD-MTSPB3. Furthermore, if $L = 0$ and $K = \sum_{i \in N'} n_i$, then Constraints (6.2)- (6.3), (6.14), (6.25)-(6.27), and (6.28)-(6.31) represent the formulation presented in Aguayo et al.(2015) for the non-fixed destination high multiplicity multi-depot traveling salesman problem.

6.3 Benders decomposition

In this section, we present Benders decomposition-based approaches for FD-MTSPB2, FD-MTSPB3, and NF-MTSPB. Note that such an approach for FD-MTSPB1 has been presented in Bektaş (2012).

6.3.1 Decomposition of FD-MTSPB2

The Benders subproblem is formulated as follows, for any given x_{ij}-variables:

$$\textbf{BSP(B2)}:$$

$$\text{Minimize} \sum_{i \in N} \sum_{j \in N, j \neq i} (0) y_{ij} + \sum_{i \in N} (0) k_i \tag{6.32}$$

subject to

$$(6.8),\ (6.14),\ (6.16)-(6.18),\ \text{and}\ (6.25)-(6.27).$$

Note that BSP(B2) can be separated into two subproblems, BSP1(B2) and BSP2(B2), where the former consists of the y_{ij}-variables (Constraints (6.8), (6.14), and (6.25)-(6.27)), and the latter optimizes over the k_i-variables ((6.16)-(6.18)). This yields the following:

BSP1(B2):

$$\text{Minimize}\ \ z = \sum_{i\in N}\sum_{j\in N, j\neq i}(0)y_{ij} \tag{6.33}$$

subject to

$$-y_{di} \geq -(Kx_{di}),\ \ \forall d\in D,\ \ i\in N' \qquad (\alpha_{di}) \tag{6.34}$$

$$y_{di} \geq Lx_{di},\ \ \forall d\in D,\ \ i\in N' \qquad (\rho_{di}) \tag{6.35}$$

$$-y_{ij} \geq -(K-1)x_{ij},\ \ \forall i,j\in N' \qquad (\gamma_{ij}) \tag{6.36}$$

$$\sum_{j\in N,\ j\neq i} y_{ji} - \sum_{j\in N',\ j\neq i} y_{ij} = 1,\ \ \forall i\in N' \quad (\beta_i) \tag{6.37}$$

$$y_{ij}\geq 0,\ \ \forall i,j\in N. \tag{6.38}$$

Let $\alpha_{di}\geq 0$, $\rho_{di}\geq 0$, $\gamma_{ij}\geq 0$, and β_i be the dual variables associated with Constraints (6.34)-(6.37), respectively. BSP1 is a feasibility problem, and therefore, only Benders feasibility cuts are generated, which are defined as follows:

$$\sum_{d\in D}\sum_{i\in N'}(-Kx_{di}\alpha_{di}+Lx_{di}\rho_{di}) - (K-1)\sum_{i\in N'}\sum_{j\in N'}x_{ij}\gamma_{ij} + \sum_{i\in N'}\beta_i \leq 0$$
$$\forall(\alpha_{di},\rho_{di},\gamma_{ij},\beta_i,\forall d,i,j)\in\Omega, \tag{6.39}$$

where Ω is the set of extreme rays of the polyhedron defined by the dual of BSP1(B2).

BSP2(B2) is formulated as follows:

BSP2(B2):

$$\text{Minimize}\ \ \sum_{i\in N}(0)k_i$$

subject to

$$k_d = d,\ \ d\in D \qquad (\omega_d) \tag{6.40}$$

$$k_i - k_j \geq |D| - 1 - (|D|-1)x_{ij},\ \ \forall i,j\in N,\ \ i\neq j \quad (\tau_{ij}) \tag{6.41}$$

$$k_i\geq 0,\ \ \forall i\in N. \tag{6.42}$$

Let ω_d and $\tau_{ij} \geq 0$ be the dual variables associated with Constraints (6.40) and (6.41), respectively. Again, this subproblem is a feasibility problem, and therefore, only Benders feasibility cuts are generated, which are given as follows:

$$\sum_{d \in D} d\omega_d + \sum_{i \in N} \sum_{j \in N,\ j \neq i} [(|D| - 1 - (|D| - 1)x_{ij})]\tau_{ij} \leq 0$$
$$\forall (\omega_d, \tau_{ij}, \forall d, i, j) \in \Lambda, \tag{6.43}$$

where Λ is the set of extreme rays of the polyhedron defined by the dual of BSP2(B2).

The master problem for FD-MTSPB2 is given by:

BMP(B2): Minimize (6.1); subject to (6.2)-(6.5), (6.24), (6.39), and (6.43).

The convergence of Benders decomposition can be accelerated by generating strong or nondominated Benders cuts to append to the master problem at each iteration (e.g., see Magnanti and Wong (1981), Papadakos (2008), and Sherali and Lunday (2013)). Accordingly, we implemented the strategy suggested in Sherali and Lunday (2013), which uses a suitable preemptive perturbation of the right-hand-side of the Benders subproblem (BSP) in order to generate maximal nondominated Benders cuts. However, the results obtained with and without the consideration of such nondominated cuts were comparable, likely because we are only generating feasibility cuts in the present context (this was the case for the other formulations considered in the sequel as well). Hence, here and below, we generate Benders cuts by directly using the dual variables produced by the solver (CPLEX).

6.3.2 Decomposition of FD-MTSPB3

Next, we present Benders decomposition for FD-MTSPB3. The Benders subproblem is given as follows:

BSP(B3) :

$$\text{Minimize} \sum_{i \in N} \sum_{j \in N, j \neq i} (0)(y_{ij} + y'_{ij}) \tag{6.44}$$

subject to

(6.8), (6.14), (6.19) – (6.23), and (6.25) – (6.27).

Note as before that BSP(B3) is separable into two subproblems denoted, BSP1(B3) and BSP2(B3), where the former optimizes over the y_{ij}-variables (Constraints (6.8), (6.14), and (6.25)-(6.27)), and the latter optimizes over the y'_{ij}-variables (Constraints (6.14), and (6.19)-(6.23)). Note that BSP1(B3) is equivalent to BSP1(B2), which was presented in the previous section, and we have:

BSP2(B3):

$$\text{Minimize} \sum_{i\in N}\sum_{j\in N, j\neq i}(0)y'_{ij}$$

subject to

$$y'_{di} = dx_{di}, \quad \forall d \in D, \quad i \in N' \qquad (\theta_{di}) \tag{6.45}$$

$$y'_{id} = dx_{id}, \quad \forall d \in D, \quad i \in N' \qquad (\lambda_{id}) \tag{6.46}$$

$$-y'_{ij} \geq -|D|x_{ij}, \quad \forall i, j \in N \qquad (\zeta_{ij}) \tag{6.47}$$

$$\sum_{j\in N,\ j\neq i} y'_{ji} - \sum_{j\in N,\ j\neq i} y'_{ij} = 0, \quad \forall i \in N' \qquad (\chi_i) \tag{6.48}$$

$$y'_{ij} \geq 0, \quad \forall i, j \in N, \quad i \neq j. \tag{6.49}$$

Let θ_{di}, λ_{id}, $\zeta_{ij} \geq 0$, and χ_i be the dual variables associated with Constraints (6.45)-(6.48), respectively. BSP2 is a feasibility problem, and therefore, only Benders feasibility cuts are generated as follows:

$$\sum_{d\in D}\sum_{i\in N'} d(x_{di}\theta_{di} + x_{id}\lambda_{id}) - \sum_{i\in N}\sum_{j\in N}|D|x_{ij}(\zeta_{ij}) \leq 0$$
$$\forall(\theta_{di}, \lambda_{id}, \zeta_{ij}, \chi_i, \forall d, i, j) \in \Upsilon, \tag{6.50}$$

where Υ is the set of extreme rays of the polyhedron defined by the dual of BSP2(B3).

Finally, the master problem for FD-MTSPB3 is given as follows:

BMP(B3): Minimize (6.1); subject to (6.2)-(6.5), (6.24), (6.39), and (6.50).

6.3.3 Decomposition of NF-MTSPB

The Benders subproblem for NF-MTSPB is equivalent to BSP1(B2), and its master problem is given as follows:

BMP(NF): Minimize (6.1); subject to (6.2)-(6.5), (6.24), (6.39).

6.4 A two-phase solution approach for the FD-MTSPB

Motivated by the fact that NF-MTSPB is a relaxation of FD-MTSPB, we present a solution approach that relies on the solution to NF-MTSPB for effectively solving FD-MTSPB to optimality. A flowchart for the proposed approach is presented in Figure 6.2. In Phase-I, Steps 1-8, we solve NF-MTSPB, and, simultaneously, we generate feasible solutions for FD-MTSPB. In Step 1, we initialize the different parameters, where z_{NF}, z^{lb}_{NF}, z, and z', respectively, indicate the objective of a feasible solution to NF-MTSPB, the best lower bound for NF-MTSPB, the objective function value of a feasible solution to FD-MTSPB found from that to NF-MTSPB as described below, and the objective value of the best known solution to FD-MTSPB found during the overall approach. In Step 2, we solve NF-MTSPB via Benders decomposition until an optimal solution is obtained. This optimization process is monitored at all times (denoted by Step 3), and when a new incumbent solution to NF-MTSPB is detected, a feasible solution to FD-MTSPB is generated (Step 4). The generation of this feasible solution is derived by solving FD-MTSPB1, FD-MTSPB2, or FD-MTSPB3, with the addition of the following constraints:

$$x_{ij} = \bar{x}_{ij}, \quad \forall i \in N, j \in N'; \tag{6.51}$$

where $\bar{x}_{ij}$ denotes the solution value for each arc (i, j), $i \in N$, $j \in N'$, in the incumbent solution to NF-MTSPB that is identified at Step 3. Note that we fix all the arc values obtained for the relaxed problem except for the last arc of each route, which is altered so that a tour starts and ends at the same depot $d \in D$. In Steps 5 and 6, we update the best known feasible solution for FD-MTSPB that is obtained while solving NF-MTSPB. The optimization process continues until an optimal solution for NF-MTSPB is at hand. Once this occurs, we go to Step 8 where we check if the best lower bound of NF-MTSPB (z^{lb}_{NF}) is equal to the best feasible solution value attained for FD-MTSPB (z'). If $z^{lb}_{NF} = z'$, then we go to Step 10 and stop. Otherwise, we proceed to Phase-II (Step 9), where FD-MTSPB1, FD-MTSPB2, or FD-MTSPB3 is solved using z' as an initial solution. Once this model is solved to optimality or the maximum time limit is reached, we go to Step 10 and stop.

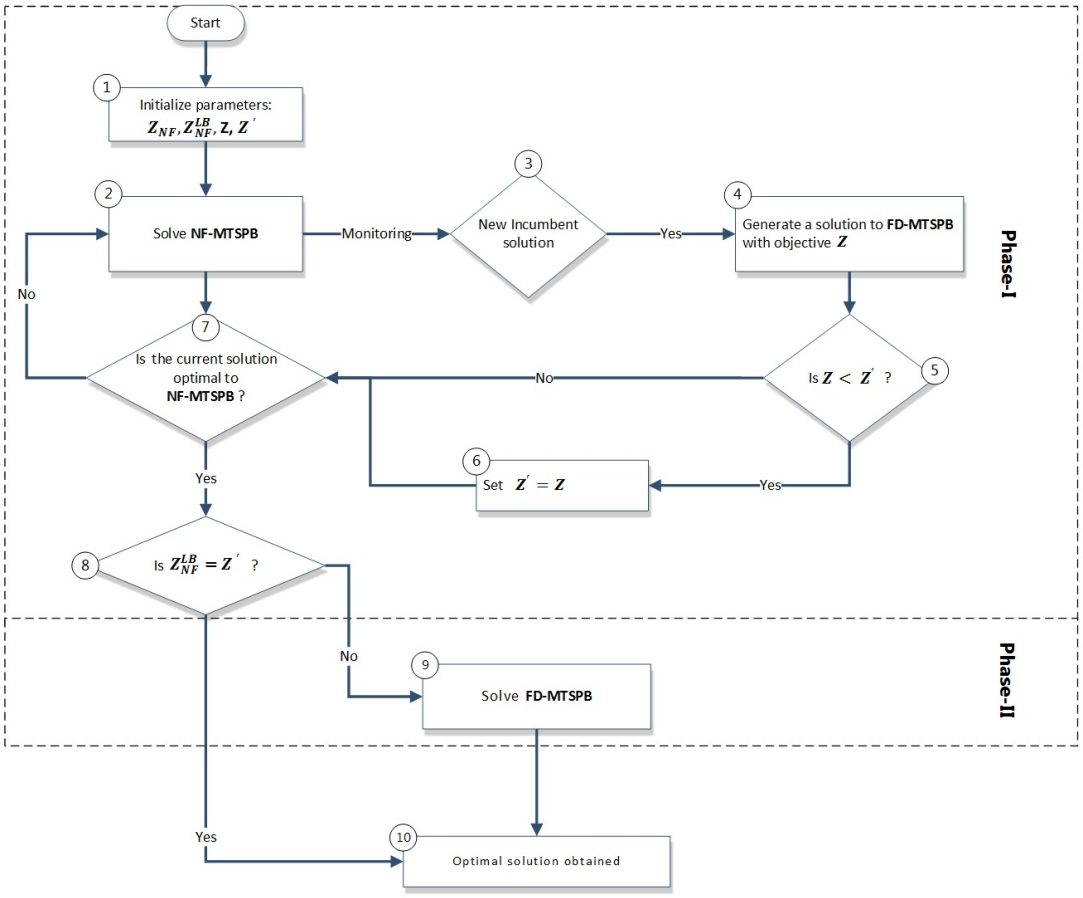

Figure 6.2: Flowchart of the proposed two-phase approach.

For illustrative purposes, we apply our proposed approach to the instance described in Figure 6.3. In this example we have, $n = 5$ (cities), $L = 1$, $K = 2$, and cities 1 and 2 are depots (squares) at each of which we have a unique salesman ($m_1 = 1$ and $m_2 = 1$). The cost matrix (designated row-based) is given by $[c_{i,j}]$={ 2, 5, 7, 8; 4, 7, 8, 8 ; 4 8, 7, 8 ; 1, 4, 0, 8; 1, 7, 8, 6}. For this instance, an incumbent solution is found for NF-MTSPB at Step 3 having an objective value $z_{NF} = 24$ (see Figure 6.3 (a)). At this point, the best lower bound value $z_{NF}^{lb} = 22$. We go to Step 4, where a feasible solution to FD-MTSPB is generated from this incumbent solution with objective value $z = 27$ (see Figure 6.3 (b)). In Steps 5 and 6, we update the best solution generated for FD-MTSPB by setting $z' = z = 27$. At Step 7, we get $z_{NF} = 24$ and $z_{NF}^{lb} = 22$, and so we continue with Step 2. Eventually, $z_{NF}^{lb} = 24$, and thus, the solution with objective value $z_{NF} = 24$ is detected as

being optimal. At Step 8, since $z^{lb}_{NF} = 24 < z' = 27$, we go to Step 9, where we use the solution having the objective $z' = 27$ as an initial solution to solve FD-MTSPB3. The final optimal solution attained at Step 9 has the objective value $z' = 25$, and is displayed in Figure 6.3 (c).

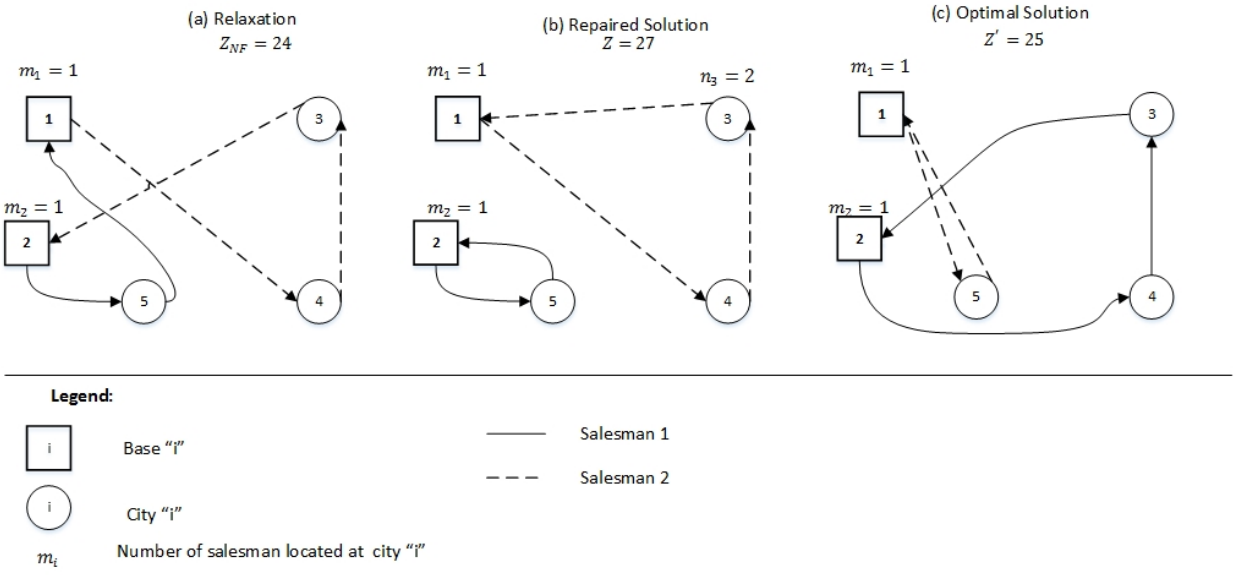

Figure 6.3: Illustration of the proposed two-phase approach.

6.5 Computational investigation

In this section, we test the performances of the three formulations presented in Section 6.2, the Benders decomposition approaches presented in Section 6.3, and the two-phase approach described in Section 6.4. For FD-MTSPB, we compare our results with those obtained by Bektaş (2012) the 63 test instances used by the latter, and for HMFD-MTSPB, we use 40 test problem instances derived from TSPLIB (1997), and compare the results obtained by solving the HMFD-MTSPB formulations directly by CPLEX with those obtained by using our decomposition approaches.

6.5.1 Results for FD-MTSPB

We compare the performances of formulations FD-MTSPB1, FD-MTSPB2, and FD-MTSPB3 when solved directly using CPLEX 12.5.1. We also present results on the Benders decomposition

schemes presented in Section 3 for the solution of FD-MTSPB using formulations FD-MTSPB2 and FD-MTSPB3 that are implemented as presented in Algorithms 1-4 below, and also using the two-phase approach as implemented using Algorithms 5 and 6 described below.

Algorithm 1. Benders decomposition scheme applied to FD-MTSPB2 (described in Section 6.3). The master problem is solved with default CPLEX settings.

Algorithm 2. Identical to Algorithm 1 except that in the master problem, we set the CPLEX parameter CPX_MIPEMPHASIS to 3 (Best Bound), i.e, an emphasis is placed on proving optimality by using the best bound value, and the detection of feasible solutions along the way becomes almost incidental.

Algorithm 3. Benders decomposition scheme applied to FD-MTSPB3 (presented in Section 6.3). The master problem is solved with default CPLEX settings.

Algorithm 4. Identical to Algorithm 3 except that in the master problem, we set CPX_MIPEMPHASIS to 3.

Algorithm 5. Two-phase solution approach (described in Section 6.4). In Phase-I, we solve NF-FTSPB via Benders decomposition, and, in Phase-II, we solve FD-MTSPB3 using Benders decomposition. Furthermore, we start Phase-II with all the Benders cuts and the incumbent solution that is generated in Phase-I. We use CPLEX with default setting in both phases.

Algorithm 6. Identical to Algorithm 5 except that in the master problem, we set CPX_MIPEMPHASIS to 3 in both phases.

The formulations were solved directly in OPL, and the Benders approaches as well as the two-phase approaches were coded using Microsoft Visual Studio Professional 2012. All runs were made on a 3.1 GHz computer having 1 GB of RAM. As in Bektaş (2012), the maximum allowable CPU time was set to 3 hours for all approaches.

We used the 63 instances of Bektaş (2012) as presented in Table 6.1 in order to test the performances of the three formulations and the six decomposition approaches. These instances are derived from TSPLIB (1997), where for each TSP instance, the first $|D|$ nodes are assigned as depots. Further-

more, for each instance, the number of salesmen at each depot, $d \in D$, is assumed to be two, and thus, a total of $2\,|D|$ salesmen are available. For each instance, the columns of this table display the instance number (#), the ATSP problem from where it was derived (ATSP instance), the number of cities (n), the number of depots ($|D|$), and the minimum and maximum number of visits per route (L and K, respectively). Following Bektaş (2012), instances 1-20 are used as an initial set to test the performance of the different formulations when solved using CPLEX as well as the Benders decomposition approaches. Instances 21-63 are next used to test the performance of the best method on small, medium, and larger-sized problem instances.

Table 6.2 presents results on the performances of the three formulations, FD-MTSPB1, FD-MTSPB2, and FD-MTSPB3 on instances 1-20 when solved using CPLEX. For each formulation, the respective columns present the objective value of the linear relaxation (z_{LP}), the objective value of the integer solution (z_{IP}), the linear programming (LP) relaxation gap (gap_{LP}) computed as $100 \times (z_{IP} - z_{LP})/z_{IP}\%$, and the CPU time in seconds or the integer gap as displayed by CPLEX at the end of the allowed CPU time limit ($T\backslash gap$). Note that FD-MTSPB1, FD-MTSPB2, and FD-MTSPB3 attained optimality in 16, 17, and 16 out of the 20 problem instances. For the 16 instances solved to optimality by all the three formulations, the LP gaps (gap_{LP}) on average are 5.24, 5.67, and 5.51 for FD-MTSPB1, FD-MTSPB2, and FD-MTSPB3, respectively. Thus, on average, FD-MTSPB1 appears to yield the tightest formulation among these three formulations. However, note that the differences between the largest and the smallest gap_{LP}-value was only 0.44%. All together, these formulations were able to solve 17 instances to optimality, and on these instances, FD-MTSPB1, FD-MTSPB2, and FD-MTSPB3 were the fastest to reach optimality in 0, 15, and 2 problem instances, respectively. Therefore, FD-MTSPB2 outperformed the other formulations in terms of both the number of instances solved to optimality and in the number of instances solved in minimum time (we have highlighted in bold the minimum T\gap value for each instance).

Table 6.3 presents results obtained for the Benders decomposition-based Algorithms 1, 2, 3, and 4 on instances 1-20. The column designations are identical to those in Table 2. Note that in column z_{IP}, " -" is used to denote the inability of an approach to generate a feasible solution for that instance. Algorithms 1, 2, 3, and 4 attained optimality in 13, 15, 15, and 16 out of the 20 problem

Table 6.1: Problem instances (Bektaş (2012))

#	*ATSP instance*	n	D	L	K	#	*ATSP instance*	n	D	L	K
1	ftv33.tsp	34	2	8	5	33	ftv64.tsp	65	2	20	5
2	ftv35.tsp	36	2	10	5	34	ftv64.tsp	65	2	20	6
3	ftv38.tsp	39	2	10	5	35	ftv64.tsp	65	2	20	7
4	p43.tsp	43	2	15	5	36	ftv64.tsp	65	2	20	8
5	ftv44.tsp	45	2	15	5	37	ftv64.tsp	65	2	30	5
6	ftv47.tsp	48	2	20	5	38	ftv64.tsp	65	2	30	6
7	ry48p.tsp	48	2	15	5	39	ftv64.tsp	65	2	30	7
8	ft53.tsp	53	2	15	5	40	ftv64.tsp	65	2	30	8
9	ftv55.tsp	56	2	15	5	41	ftv64.tsp	65	2	40	5
10	ftv55.tsp	56	3	10	5	42	ftv64.tsp	65	2	40	6
11	ftv64.tsp	65	2	20	5	43	ftv64.tsp	65	2	40	7
12	ftv64.tsp	65	3	30	5	44	ftv64.tsp	65	2	40	8
13	ft70.tsp	70	2	20	5	45	ftv64.tsp	65	2	50	5
14	ft70.tsp	70	3	15	5	46	ftv64.tsp	65	2	50	6
15	ftv70.tsp	71	2	30	5	47	ftv64.tsp	65	2	50	7
16	ftv70.tsp	71	3	20	5	48	ftv64.tsp	65	2	50	8
17	kro124p.tsp	100	2	40	5	49	kro100.tsp	100	2	30	5
18	kro124p.tsp	100	3	35	5	50	kro100.tsp	100	2	35	5
19	ftv170.tsp	171	5	40	5	51	kro100.tsp	100	2	40	5
20	ftv170.tsp	171	5	50	5	52	kro100.tsp	100	3	30	5
21	ftv38.tsp	39	2	20	5	53	kro100.tsp	100	3	35	5
22	ftv38.tsp	39	2	20	6	54	kro100.tsp	100	3	40	5
23	ftv38.tsp	39	2	20	7	55	ftv64.tsp	65	2	6	16
24	ftv38.tsp	39	2	20	8	56	ftv64.tsp	65	2	7	17
25	ftv38.tsp	39	2	30	5	57	ftv64.tsp	65	2	8	18
26	ftv38.tsp	39	2	30	6	58	ftv64.tsp	65	2	5	18
27	ftv38.tsp	39	2	30	7	59	ftv64.tsp	65	2	6	19
28	ftv38.tsp	39	2	30	8	60	ftv64.tsp	65	2	7	20
29	ftv38.tsp	39	2	40	5	61	ftv38.tsp	38	2	6	50
30	ftv38.tsp	39	2	40	6	62	ftv38.tsp	38	2	7	40
31	ftv38.tsp	39	2	40	7	63	ftv38.tsp	38	2	8	30
32	ftv38.tsp	39	2	40	8						

Table 6.2: Comparison of the solutions for formulations FD-MTSPB1, FD-MTSPB2, and FD-MTSPB3 obtained using CPLEX on instances 1-20.

Ins	***FD-MTSPB1***				***FD-MTSPB2***				***FD-MTSPB3***			
#	z_{LP}	z_{IP}	gap_{LP}	$T\backslash gap$	z_{LP}	z_{IP}	gap_{LP}	$T\backslash gap$	z_{LP}	z_{IP}	gap_{LP}	$T\backslash gap$
1	1425.17	1579	9.74%	149.31	1422.65	1579	9.90%	**106.75**	1424.33	1579	9.80%	132.30
2	1580.48	1669	5.30%	13.17	1581.09	1669	5.27%	**10.81**	1581.33	1669	5.25%	18.86
3	1635.87	1730	5.44%	68.69	1634.90	1730	5.50%	**13.71**	1635.14	1730	5.48%	29.34
4	1979.98	5695	65.23%	0.05%	2055.86	5695	63.90%	**329.08**	2055.86	5695	63.90%	0.03%
5	1705.08	1802	5.38%	63.57	1705.08	1802	5.38%	**14.90**	1705.08	1802	5.38%	43.46
6	1855.97	1975	6.03%	139.37	1832.41	1975	7.22%	**114.47**	1848.25	1975	6.42%	234.94
7	15049.37	15684	4.05%	278.87	15049.37	15684	4.05%	**103.32**	15049.37	15864	4.05%	214.91
8	6792.02	7396	8.17%	57.07	6707.06	7396	9.32%	139.15	6730.71	7396	9.00%	**44.65**
9	1650.73	1797	8.14%	851.38	1642.08	1797	8.62%	**658.12**	1642.58	1797	8.59%	1164.27
10	1877.48	2013	6.73%	328.35	1863.12	2013	7.45%	**53.46**	1865.66	2013	7.32%	114.74
11	1890.46	1992	5.10%	277.53	1882.86	1992	5.48%	**172.19**	1887.44	1992	5.25%	678.65
12	1939.25	2062	5.95%	5877.13	1893.36	2062	8.18%	**848.54**	1908.12	2062	7.46%	1954.55
13	40789.50	41105	0.77%	2907.11	40777.60	41105	0.80%	**319.76**	40780.20	41105	0.79%	1078.93
14	41704.20	42272	1.34%	9690.80	41703.39	42272	1.35%	5765.86	41703.39	42272	1.35%	**2415.11**
15	1967.02	2074	5.16%	213.50	1953.88	2074	5.79%	**55.88**	1960.06	2074	5.49%	124.19
16	2090.41	2326	10.13%	3.71%	2083.24	2296	9.27%	0.65%	2084.31	2296	9.22%	**0.62%**
17	36311.99	37407	2.93%	1301.92	36311.99	37407	2.93%	**403.66**	36311.98	37407	2.93%	489.11
18	36720.99	38076	3.56%	1375.84	36715.78	38076	3.57%	**431.31**	36719.7	38076	3.56%	1119.71
19	3121.42	26568	88.25%	88.07%	3120.65	19650	84.12%	83.32%	3120.65	5863	46.77%	**44.33%**
20	3094.80	27044	88.56%	88.40%	3091.48	5085	39.20%	**35.72%**	3091.80	23393	86.78%	86.01%

instances, respectively. All together, these Benders decomposition-based approaches were able to solve 17 instances to optimality, and on these instances, Algorithms 1, 2, 3, and 4 were the fastest in 0, 2, 5, and 10 problem instances, respectively. Therefore, Algorithm 4 outperformed the others in terms of both the number of instances solved to optimality and the number of instances solved in minimum time. The minimum T\gap value is highlighted in bold for each instance.

Table 6.4 displays the results obtained for the different implementations of the two-phase approach (Algorithms 5 and 6) on the initial set of 20 instances. The table columns represent the value of the optimal solution or best bound obtained for NF-MTSPB in Phase-I (z_{NF}), the objective value of the initial solution generated for FD-MTSPB in Phase-I (z_{IS}), the objective value of the integer solution obtained by the overall approach (z_{IP}), the CPU time (in seconds) taken to solve NF-MTSPB (T_{NF}), and the *T/gap*, respectively. Again, " -' is used to denote the inability of an approach to generate a feasible solution for that instance. First, note that the initial solution value (z_{IS}) was very close to the true optimal objective value of FD-MTSPB. The initial solutions generated for Algorithms 5 and 6 were optimal to FD-MTSPB (i.e, $z_{IS} = z_{IP}$) in 6 and 7 out of 20 problem instances, respectively. Also, the minimum, maximum, and average integer gaps

Table 6.3: Comparison of Benders decomposition-based Algorithms 1, 2, 3, and 4 on instances 1-20.

Ins	**Alg. 1**		**Alg. 2**		**Alg. 3**		**Alg. 4**	
#	z_{IP}	$T\backslash gap$	z_{IP}	$T\backslash gap$	z_{IP}	$T\backslash gap$	z_{IP}	$T\backslash gap$
1	1579	289.43	1579	26.88	1579	20.0	1579.00	**17.3**
2	1669	76.39	1669	5.27	1669	**4.7**	1669	7.0
3	1730	110.78	1730	13.17	1730	**5.2**	1730	9.7
4	5695	2513.19	5695	**1752.67**	5695	2241.7	5695	2404.5
5	1802	784.37	1802	30.03	1802	81.3	1802	**15.0**
6	1975	377.83	1975	443.27	1975	155.2	1975	**127.7**
7	15864	106.00	15864	48.11	15864	67.2	15684	**35.6**
8	7396	302.19	7396	387.93	7396	75.7	15864	**13.7**
9	1797	3824.16	1797	1422.84	1797	1197.5	1797	**819.4**
10	-	3 hours	2013	178.33	2013	**100.4**	2013	1152.7
11	1992	2590.9	1992	322.55	1992	**513.4**	1992	929.3
12	2083	3.34%	-	3 hours	-	3 hours	2062	**3927.7**
13	41105	1689.12	41105	552.52	41105	741.1	41105	**222.9**
14	43369	2.97%	-	3 hours	42272	3 hours	42272	**7417.6**
15	2074	693.50	2074	197.51	2074	303.2	2074	**44.2**
16	3674	**39.52%**	-	3 hours	-	3 hours	-	3 hours
17	37407	7196.36	37407	403.36	37407	**386.9**	-	3 hours
18	38076	1.34%	38076	**5761.12**	38076	8754.12	38076	6754.12
19	-	3 hours	-	3 hours	-	3 hours	-	3 hours
20	-	3 hours	-	3 hours	-	3 hours	-	3 hours

(computed as $100 * (z_{IS} - z_{IP})/z_{IP}\%$), were 0.00%, 10.17%, and 1.34% for Algorithm 5, and 0.00%, 10.05%, and 1.49% for Algorithm 6, respectively. Overall, Algorithms 5 and 6 solved to optimality 16 and 17 of the 20 problem instances, respectively, and for the 16 problem instances that both algorithms solved to optimality, Algorithm 6 was the fastest in 12 cases, thus outperforming Algorithm 5 in terms of both the number of instances solved to optimality and number of instances solved in minimum time. As before, we have highlighted in bold the minimum *T/gap* value, for each instance.

Table 6.4: Comparison of different implementations of the proposed two-phase approach (Algorithm 5 and 6) on instances 1-20.

Ins	**Alg.5**					**Alg.6**				
#	z_{NF}	z_{IS}	z_{IP}	T_{NF}	$T\backslash gap$	z_{NF}	z_{IS}	z_{IP}	T_{NF}	$T\backslash gap$
1	1563	1579	1579	20.67	23.30	1563	1579	1579	20.3	**22.94**
2	1656	1672	1669	14.52	15.11	1656	1672	1669	2.1	**2.16**
3	1730	1770	1730	14.88	15.69	1726	1770	1730	7.1	**7.83**
4	5695	5695	5695	2064.12	2064.12	5695	5695	5695	1850.1	**1850.12**
5	1802	1805	1802	49.69	**57.22**	1802	1842	1802	64.9	66.59
6	1936	2016	1975	16.67	**39.98**	1936	2016	1975	9.8	44.69
7	15812	17459	15684	105.425	108.26	15812	17459	15864	33.5	**47.65**
8	7213	7562	7396	3.18	**30.85**	7213	7562	7396	2.9	61.58
9	1797	1797	1797	4572.89	4572.89	1797	1797	1797	1137.9	**1137.93**
10	2008	2013	2013	3 hours	0.25%	2013	2013	2013	52.6	**52.60**
11	1992	1996	1992	590.542	604.79	1992	1996	1992	126.1	**133.91**
12	2053	2144	2062	737.23	2737.57	2053	2144	2062	677.2	**2345.12**
13	41075	41360	41105	611.82	685.95	41075	41155	41105	209.4	**142.43**
14	42067	42274	42272	111.08	**3089.62**	42067	43113	42272	25.8	5094.63
15	2070	2098	2074	219.32	223.44	2070	2074	2074	4.8	**105.36**
16	2207	2417	2417	3 hours	**8.69%**	2262	-	-	3 hours	3 hours
17	37407	37407	37407	7640.57	7640.57	37407	37407	37407	339.4	**339.43**
18	38073	38076	38076	5787.86	5821.88	38073	38076	38076	4980.9	**5002.87**
19	3166	5367	5367	3 hours	**41.02%**	3227.3	-	-	3 hours	3 hours
20	3124	-	-	3 hours	3 hours	3218	-	-	3 hours	3 hours

Table 6.5 presents a comparison of the computational effort required by the most effective formulation (FD-MTSPB2) solved by CPLEX, versus that for the proposed most effective Benders decomposition-based approach (Algorithm 4) and the most effective two-phase approach (Algorithm 6). The CPU times required to solve each instance to optimality by CPLEX applied to FD-MTSPB2, and Algorithms 4 and 6 are presented in columns 2, 3, and 4, respectively. CPLEX

and Algorithm 4 solved to optimality 17 and 16 out of 20 problem instances, respectively, and for the 16 instances that both solved to optimality, CPLEX was the fastest in 9 out 16 cases, thus, outperforming Algorithm 4 in terms of both the number of instances solved to optimality and the number of instances solved in minimum time. Between CPLEX and Algorithm 6, both attained optimal solutions for 17 out of 20 problem instances, but for the 17 problem instances solved to optimality, Algorithm 6 was the fastest in 11 out of 17 cases. Thus, Algorithm 6 outperformed both CPLEX and Algorithm 4 in terms of the number of instances solved to optimality (equalling CPLEX in this case) and instances solved in minimum time.

Table 6.5: Comparison of CPU times for the most effective formulation (FD-MTSPB2) solved directly by CPLEX versus the proposed most effective Benders approach (Algorithm 4), and the most effective two-phase approach (Algorithm 6) on instances 1-20.

#	CPLEX	Alg. 4	Alg. 6
1	106.75	**17.3**	22.94
2	10.81	7.0	**2.16**
3	13.71	9.7	**7.83**
4	**329.08**	2404.5	1850.12
5	**14.90**	15.0	66.59
6	114.47	127.7	**44.69**
7	103.32	**35.6**	47.65
8	139.15	**13.7**	61.58
9	**658.12**	819.4	1137.93
10	53.46	1152.7	**52.60**
11	172.19	929.3	**133.91**
12	**848.54**	3927.7	2345.12
13	319.76	222.9	**142.43**
14	5765.9	7417.6	**5094.63**
15	55.88	**44.2**	105.36
16	3 hours	3 hours	3 hours
17	403.66	3 hours	**339.43**
18	**431.31**	6754.12	5002.87
19	3 hours	3 hours	3 hours
20	3 hours	3 hours	3 hours

We point out that on the 17 instances that both CPLEX and Algorithm 6 solved to optimality, the average CPU time for Algorithm 6 (933.18 seconds) was almost twice the time taken by CPLEX (552.48 seconds). However, the average time for the former approach is skewed mainly because of

the excessive CPU times consumed for two instances: 12 and 18 (the average times without these instances for Algorithm 6 and CPLEX were 550.74 and 607.32, respectively, which are comparable). Nonetheless, our aim has been to develop an optimal seeking approach for FD-MTSPB, and thus, our metric for comparison, similar to that used by Bektaş (2012), has been the number of instances solved to optimality and the number of instances solved in minimum time.

In view of the above results, we only present further results of using Algorithm 6 on the remaining instances (21-63). Table 6.6 displays these results, where the columns are designated as in the previous tables. Again, note that the initial solution (z_{IS}) was very close to the optimal objective value of FD-MTSPB, where the minimum, maximum, and average integer gaps of the initial solution computed as $100(z_{IS} - z_{IP})/z_{IP}\%$ are 0.00%, 5.44%, and 0.68%, respectively. Also, Algorithm 6 solved to optimality 41 out of the 43 problem instances; no feasible solutions were obtained for instances # 49 and # 50.

As for comparing the results for Algorithm 6 against the Benders-based approaches described by Bektaş (2012), we first of all acknowledge the difficulty in making a fair comparison due to the following. First, Bektaş (2012) used a 3 GHz computer and 1 GB of RAM along with AMPL/CPLEX suite version 11.2 implemented using default parameters, whereas we use a 3.1 GHz computer having 1 GB of RAM along with Microsoft Visual Studio Professional 2012 with CPLEX (12.5.1) implemented using default parameters, but with MIP emphasis switched to Best Bound in some cases. Second, there does not exist a unified approach to implement a Benders decomposition scheme. Nevertheless, next, we present in Table 6.7 a comparison of CPU times consumed by the different Benders algorithms developed in Bektaş (2012) and Algorithm 6. Bektaş (2012) developed seven Benders-based approaches, B_u, B_w, B_{dw}, B_s, B_{ds}, B_m, and B_{dm}, but the results for the 63 test problem instances are reported only for the best approaches (B_u, B_w, and B_{dw}). The procedures B_u, B_w, B_{dw}, and Algorithm 6 attained optimality in 35, 57, 57, and 60 out of the 63 instances, and were the fastest in 4, 12, 5, and 38 instances, respectively. Thus, Algorithm 6 outperformed the others in both the number of solutions solved to optimality and the number of problems solved in minimum time.

Overall, it can be seen that our proposed two-phase approach, Algorithm 6, is quite effective in

Table 6.6: Computational results for the two-phase approach (Algorithm 6) on instances 21-63.

	Alg. 6						**Alg. 6**				
#	Z_{NF}	Z_{IS}	z_{IP}	B_{nf}	$T\backslash gap$	#	Z_{NF}	Z_{IS}	z_{IP}	B_{nf}	$T\backslash gap$
21	1654	1658	1658	2.75	3.56	43	1992	2020	1992	4543.2	5018.82
22	1682	1710	1702	55.34	65.09	44	1992	2020	1992	2975.06	3401.27
23	1721	1749	1730	678.32	1169.54	45	1961	1997	1968	38.38	76.79
24	1726	1730	1730	400.67	1334.07	46	1979	1992	1992	1652.72	2403.77
25	1654	1658	1658	2.71	3.15	47	1992	1996	1992	3469.28	3719.44
26	1682	1710	1702	26.69	44.5	48	1992	1992	1992	2699.18	2699.18
27	1721	1749	1730	580.56	1882.12	49	37223	-	-	3 hours	3 hours
28	1726	1749	1730	904.14	1638.49	50	37573	-	-	3 hours	3 hours
29	1654	1658	1658	2.71	3.02	51	37407	37407	37407	333.69	333.69
30	1682	1710	1702	20.59	52.35	52	37709	40407	38322	2682.76	3075.33
31	1721	1731	1730	1123.27	2270.58	53	38073	38076	38076	5428.79	5454.12
32	1726	1770	1730	984.71	2052.32	54	37842	37844	37844	369.08	371.22
33	1992	1996	1992	125.32	133.05	55	2036	2088	2050	206.20	433.96
34	1992	2020	1992	356.90	374.02	56	2008	2023	2017	87.43	210.20
35	1992	2008	1992	215.47	230.07	57	2008	2017	2014	125.33	173.20
36	1992	1996	1974	194.52	202.18	58	2008	2017	2014	125.33	173.20
37	1961	2001	1992	62.50	114.26	59	1992	1996	1996	89.01	92.88
38	1979	1996	1992	1036.17	1349.71	60	1992	1992	1992	225.25	225.25
39	1992	1996	1992	1035.26	1794.85	61	1682	1702	1702	22.75	47.95
40	1992	1996	1992	3167.24	3404.38	62	1721	1731	1730	741.25	1352.48
41	1961	2005	1974	96.56	171.08	63	1726	1749	1730	902.37	1635.91
42	1979	2014	1992	525.08	1446.33						

Table 6.7: Comparison of the Benders-based algorithms B_u, B_w, and B_{dw} presented in Bektaş (2012) and the proposed two-phase approach (Algorithm 6).

#	B_u	B_w	B_{dw}	*Alg. 6*	#	B_u	B_w	B_{dw}	*Alg. 6*
1	4281.2	220.1	91.8	**22.94**	33	3 hours	361.3	776.7	**133.05**
2	67.5	7.3	4.9	**2.16**	34	3 hours	**232.6**	530.0	374.02
3	350.5	13.9	17.6	**7.83**	35	3 hours	340.3	540.3	**230.07**
4	3 hours	**120.5**	718.5	1850.12	36	3 hours	420.7	208.4	**202.18**
5	46.7	39.1	**39.0**	66.59	37	2443.8	205.6	170.7	**114.26**
6	767.9	271.0	117.2	**44.69**	38	3 hours	1770.4	**824.2**	1349.71
7	3 hours	32.0	**27.3**	47.65	39	3 hours	**771.5**	1029.6	1794.85
8	9999.0	**14.7**	17.0	61.58	40	3089.9	1135.1	**755.6**	3404.38
9	3 hours	3987.2	3994.6	**1137.93**	41	5122.2	**148.1**	186.1	171.08
10	3 hours	362.3	2584.6	**52.60**	42	3 hours	**975.1**	1193.2	1446.33
11	3 hours	340.0	722.0	**133.91**	43	3 hours	**1156.8**	1207.3	5018.82
12	3513.5	**1863.9**	2010.5	2345.12	44	2726.2	**808.2**	1132.3	3401.27
13	481.3	441.9	1357.2	**142.43**	45	1039.8	117.0	98.1	**76.79**
14	3 hours	3 hours	7416.9	**5094.63**	46	3 hours	**941.6**	1130.6	2403.77
15	543.2	120.1	118.4	**105.36**	47	3 hours	**641.8**	1456.8	3719.44
16	3 hours	3 hours	3 hours	3 hours	48	3 hours	**830.3**	1254.9	2699.18
17	740.6	5228.4	8236.7	**339.43**	49	3 hours	3 hours	3 hours	3 hours
18	**3292.0**	3955.7	5044.9	5002.87	50	**3851.0**	7668.6	3 hours	3 hours
19	3 hours	3 hours	3 hours	3 hours	51	740.6	5228.5	8236.7	**333.69**
20	3 hours	3 hours	3 hours	3 hours	52	3 hours	3 hours	3 hours	**3075.33**
21	6.6	4.0	4.0	**3.56**	53	**3292.0**	3955.7	5044.9	5454.12
22	299.2	78.4	171.7	**65.09**	54	421.6	2166.9	5178.1	**371.22**
23	**1079.6**	4158.5	3297.0	1169.54	55	3 hours	4620.6	9599.7	**433.96**
24	1867.7	9349.3	4155.2	**1334.07**	56	3 hours	438.1	514.2	**210.20**
25	5.2	4.1	3.9	**3.15**	57	3 hours	593.6	1522.9	**173.20**
26	187.7	112.2	172.9	**44.50**	58	3 hours	463.2	717.3	**173.20**
27	3025.9	3848.3	3345.4	**1882.12**	59	3 hours	546.2	360.0	**92.88**
28	1900.7	9352.7	8181.8	**1638.49**	60	3 hours	340.3	540.3	**225.25**
29	9.9	4.4	**2.9**	3.02	61	152.2	138.4	137.9	**47.95**
30	197.6	130.9	126.3	**52.35**	62	5614.6	4253.7	3352.5	**1352.48**
31	5614.6	4253.7	3352.5	**2270.58**	63	1900.7	9352.7	8181.8	**1635.91**
32	2379.1	8312.3	5599.4	**2052.32**					

solving FD-MTSPB. This approach is also versatile in that Phase-I can be implemented using a Benders decomposition scheme such as the one presented in Section 6.3 or it can be implemented directly using an off-the-shelf software (e.g, CPLEX). Similarly, Phase-II can be executed either using any of Benders decomposition frameworks described in this chapter (or the ones available in the literature) or it can be implemented directly using a commercial software. Furthermore, we point out that a user can avail of parallel computing environments in Phase-I and Phase-II when implementing these using a suitable commercial software such as CPLEX, as opposed to the specialized Benders decomposition-based approaches adapted herein.

6.5.2 Results for the high-multiplicity FD-MTSPB

Both HMFD-MTSPB1 and HMFD-MTSPB2 formulations were solved directly with CPLEX (12.5.1) using OPL with default parameters and with a time limit of three-hours of CPU time. For test purposes, we developed 40 problem instances for HMFD-MTSPB, designated as #64-103 in Tables 6.8 and 6.9. These were derived from instances 1-20 presented in the previous section, where the number of visits required for each city, n_i, was obtained using uniform distributions U[1,5] and U[6,10], and the interval $[L, K]$ was chosen arbitrarily in order to generate both non-trivial instances and feasible solutions. Furthermore, we also assumed that the number of salesmen at each depot is two, and thus, the total number of salesmen is $2|D|$.

Table 6.8 presents a comparison between the performance of CPLEX when directly applied to solve formulations HMFD-MTSPB1 and HMFD-MTSPB2 on instances 64-83. Both formulations solved all instances to optimality except for two problems derived from the ATSP instance p43 (instances # 70 and #71). For instance # 70, the optimality gaps for HMFD-MTSPB1 and HMFD-MTSPB2 were 61.05% and 60.94%, respectively, while for instance #71, the optimality gaps for HMFD-MTSPB1 and HMFD-MTSPB2 were 0.45% and 0.27%, respectively. For the 18 instances solved to optimality by both formulations, HMFD-MTSPB1 and HMFD-MTSPB2 were solved fastest in 4 and 14 cases, respectively. On average, HMFD-MTSPB2 offered a somewhat tighter LP relaxation than HMFD-MTSPB1 (on average, the LP gaps were 7.88% and 7.96%, respectively). We have highlighted the minimum $T\backslash gap$, for each instance.

Table 6.8: Comparison of the direct solution of formulations HMFD-MTSPB1 and HMFD-MTSPB2 by CPLEX on instances 64-83.

Instances								HMFTSPB1				HMFTSPB3			
#	*ATSP instance*	n	D	n_i	$\sum_{i \in N} n_i$	L	K	z_{LP}	z_{IP}	gap_{LP}	$T\backslash gap$	z_{LP}	z_{IP}	gap_{LP}	$T\backslash gap$
64	ftv33.tsp	34	2	U[1,5]	86	10	30	3369.45	3382	0.37%	**0.34**	3369.45	3382	0.37%	0.47
65	ftv33.tsp	34	2	U[6,10]	259	20	70	9924.96	9949	0.24%	0.17	9924.96	9949	0.24%	**0.16**
66	ftv35.tsp	36	2	U[1,5]	104	10	30	4826.45	4847	0.42%	0.97	4826.12	4847	0.43%	**0.84**
67	ftv35.tsp	36	2	U[6,10]	276	20	70	11530.43	11538	0.07%	**0.55**	11529.03	11538	0.08%	0.67
68	ftv38.tsp	39	2	U[1,5]	104	10	30	4717.55	4740	0.47%	0.45	4717.55	4740	0.47%	**0.31**
69	ftv38.tsp	39	2	U[6,10]	288	20	80	10838.82	10849	0.09%	0.25	10838.82	10849	0.09%	**0.20**
70	p43.tsp	43	2	U[1,5]	125	10	40	2031.72	5967	65.95%	61.05%	2031.72	5697	64.34%	**60.94%**
71	p43.tsp	43	2	U[6,10]	308	20	160	1401.81	5638	75.14%	0.45%	1407.77	5638	75.03%	**0.27%**
72	ftv44.tsp	45	2	U[1,5]	124	10	40	5108.00	5108	0.00%	0.75	5108.00	5108	0.00%	**0.20**
73	ftv44.tsp	45	2	U[6,10]	339	30	90	12300.00	12300	0.00%	**0.19**	12300.00	12300	0.00%	0.25
74	ftv47.tsp	48	2	U[1,5]	136	10	40	5818.90	5851	0.55%	1.29	5811.494	5851	0.68%	**1.26**
75	ftv47.tsp	48	2	U[6,10]	370	30	100	13611.68	13628	0.12%	**0.64**	13611.68	13628	0.12%	0.66
76	ry48p.tsp	48	2	U[1,5]	144	10	40	40804.06	41519	1.72%	5.32	40804.06	41519	1.72%	**4.77**
77	ry48p.tsp	48	2	U[6,10]	370	30	100	100224.67	100348	0.12%	0.67	100224.67	100348	0.12%	**0.31**
78	ft53.tsp	53	2	U[1,5]	157	10	40	20741.39	21188	2.11%	54.04	20741.39	21188	2.11%	**8.61**
79	ft53.tsp	53	2	U[6,10]	424	30	120	50197.05	51364	2.27%	0.84	50197.05	51364	2.27%	**0.72**
80	ftv55.tsp	56	2	U[1,5]	157	10	40	5142.67	5173	0.59%	1.92	5142.67	5173	0.59%	**1.20**
81	ftv55.tsp	56	2	U[6,10]	427	30	120	11909.41	11954	0.37%	1.59	11909.41	11954	0.37%	**1.26**
82	ftv55.tsp	56	3	U[1,5]	157	10	30	5146.49	5178	0.61%	3.21	5146.49	5178	0.61%	**1.78**
83	ftv55.tsp	56	3	U[6,10]	421	30	80	11931.27	11972	0.34%	2.04	11931.27	11972	0.34%	**1.22**

Table 6.9 presents similar results using instances 84-103. Note that both formulations solved all these problem instances to optimality, where HMFD-MTSPB1 and HMFD-MTSPB2 were solved the fastest in 3 and 17 out of the 20 cases, respectively. Furthermore, both formulations yielded comparable LP relaxation gaps of 0.15% on average. The minimum $T\backslash gap$ is highlighted for each instance.

Finally, Table 6.10 displays the results for Algorithm 6 using the relatively harder instances from Tables 6.8 and 6.9 for which both formulations required more than 40 CPU seconds. The designation of Columns 2-6 is as before, and Column 7 presents, for each instance, the smaller ($T\backslash gap$) attained by CPLEX when solving HMFD-MTSPB1 or HMFD-MTSPB2. Algorithm 6 solved seven out of eight problems instances to optimality, while CPLEX only solved six of these instances to optimality. Furthermore, for the six instances solved to optimality by Algorithm 6 and CPLEX, the former attained optimal solutions faster (using either formulation) in five cases. As before, we have highlighted the minimum $T\backslash gap$, for each instance.

Table 6.9: Comparison of the direct solution of formulations HMFD-MTSPB1 and HMFD-MTSPB2 by CPLEX on instances 84-103

Instances								**HMFTSPB1**				**HMFTSPB3**			
#	*ATSP instance*	n	D	n_i	$\sum_{i\in N} n_i$	L	K	z_{LP}	z_{IP}	gap_{LP}	$T\backslash gap$	z_{LP}	z_{IP}	gap_{LP}	$T\backslash gap$
84	ftv64.tsp	65	2	U[1,5]	200	10	50	5690.00	5690	0.00%	2.28	5689.76	5690	0.00%	**0.97**
85	ftv64.tsp	65	2	U[6,10]	508	50	150	14516.17	14522	0.04%	2.64	14516.17	14522	0.04%	**1.12**
86	ftv64.tsp	65	3	U[1,5]	195	10	35	5693.47	5697	0.06%	3.46	5692.02	5697	0.09%	**1.50**
87	ftv64.tsp	65	3	U[6,10]	500	50	100	14516.77	14522	0.04%	**1.08**	14516.77	14522	0.04%	1.29
88	ft70.tsp	70	2	U[1,5]	212	10	60	120973.01	121090	0.10%	1.20	120973.01	121090	0.10%	**1.14**
89	ft70.tsp	70	2	U[6,10]	546	30	140	303225.22	303313	0.03%	1.12	303225.22	303313	0.03%	**1.09**
90	ft70.tsp	70	3	U[1,5]	209	10	35	120533.27	120587	0.04%	1.47	120533.27	120587	0.04%	**1.09**
91	ft70.tsp	70	3	U[6,10]	537	30	100	299177.04	299325	0.05%	1.78	299117.04	299235	0.04%	**0.84**
92	ftv70.tsp	71	2	U[1,5]	209	10	60	6125.93	6130	0.07%	2.11	6125.93	6130	0.07%	**1.39**
93	ftv70.tsp	71	2	U[6,10]	564	30	150	15457.66	15479	0.14%	2.20	15457.66	15479	0.14%	**2.09**
94	ftv70.tsp	71	3	U[1,5]	206	10	40	6152.97	6158	0.08%	3.34	6152.97	6158	0.08%	**1.48**
95	ftv70.tsp	71	3	U[6,10]	554	30	100	15351.85	15362	0.07%	3.40	15351.85	15362	0.07%	**1.68**
96	kro124p.tsp	100	2	U[1,5]	268	30	70	103373.81	104209	0.80%	385.74	103373.81	104209	0.80%	**91.34**
97	kro124p.tsp	100	2	U[6,10]	796	50	200	273823.47	274051	0.08%	15.37	273823.47	274051	0.08%	**12.21**
98	kro124p.tsp	100	3	U[1,5]	267	30	50	103706.24	104512	0.77%	**67.89**	103703.31	104512	0.77%	122.43
99	kro124p.tsp	100	3	U[6,10]	788	50	200	273823.47	274051	0.08%	**12.31**	273823.47	274051	0.08%	14.37
100	ftv170.tsp	171	5	U[1,5]	502	20	60	9421.07	9448	0.29%	443.84	9421.07	9448	0.29%	**66.36**
101	ftv170.tsp	171	5	U[6,10]	1310	20	140	21851	21851	0.00%	228.54	21851	21851	0.00%	**87.47**
102	ftv170.tsp	171	5	U[1,5]	502	40	100	9411.77	9437	0.27%	230.62	9411.77	9437	0.27%	**58.17**
103	ftv170.tsp	171	5	U[6,10]	1310	20	180	21851	21851	0.00%	164.11	21851	21851	0.00%	**47.61**

Table 6.10: Comparison of Algorithm 6 and the best directly solved formulation using CPLEX on the relatively harder HMFD-MTSPB problem instances from Tables 6.8 and 6.9.

	Algorithm 6					CPLEX on best formulation
#	z_{NF}	z_{IS}	z_{IP}	B_{nf}	$T\backslash gap$	$T\backslash gap$
70	5697	5697	5697	1088.40	**1088.40**	60.94%
71	5638	5638	5638	3 hours	0.32%	**0.27%**
96	104202	104209	104209	63.52	**65.06**	91.34
98	104429	107267	104512	35.4	**40.81**	67.89
100	9438	9438	9438	13.48	**13.48**	66.36
101	21851	21851	21851	12.68	**12.68**	87.47
102	9437	9437	9437	122.85	122.85	**58.17**
103	21851	21851	21851	11.87	**11.87**	47.61

Chapter 7

Vehicle Routing Problem with Temporary Storage (VRP-TS)

7.1 Introduction

In the vehicle routing problem (VRP) deliveries are made to a set of customers, each with a known demand (less than the vehicle capacity) that must be satisfied by a unique vehicle. By allowing a customer to be served by more than one vehicle gives rise to a variant known as the VRP with split delivery VRP-SD (Dror and Trudeau (1990)). The VRP-SD has been intensively studied in the literature over the last decade, and it has been shown that split delivery can be advantageous in reducing the routing cost. Moreover, a tight bound has been proven for this reduction.

We introduce an extension of the VRP, wherein customer locations can be used for temporary storage, denoted as the VRP with temporary storage (VRP-TS). Both the VRP and VRP-SD assume that customers only accept deliveries, but they overlook the possibility of customers holding temporarily extra items, which can be picked up later by another vehicle. This feature transforms a location into a temporary storage. Figure 7.1 illustrates the type of deliveries (regular, split, and temporary) that are permitted in each problem. In a regular delivery, the demand of each customer must be satisfied by exactly one vehicle, and in a split delivery, the demand of each customer might

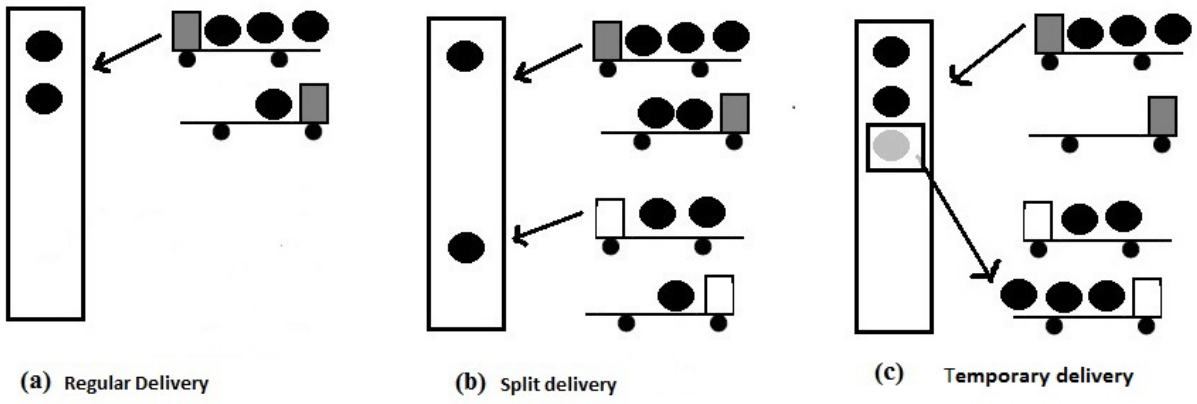

Figure 7.1: Different operations at a given customer location. (a) regular delivery: one vehicle delivers customer demand. (b) split delivery: two vehicles deliver customer demand. (c) temporary storage: one vehicle satisfies customer demand, and it drops-off 1 extra unit that is picked up later by another vehicle.

be satisfied by multiple vehicles. Meantime, in a temporary delivery, the demand of each customer must be satisfied by exactly one vehicle, but multiples vehicles can drops-off the extra material at this location that is picked up later (by possible several vehicles). The VRP only accept regular deliveries (Figure 7.1.a), and the VRP-SD allows regular deliveries and split deliveries (Figure 7.1.a and b). The VRP-TS allows regular and temporal deliveries (Figure 7.1.a, and c). In Figure 7.1.c, a vehicle drop-off 3 items, and 1 item is later picked up by another vehicle.

VRP-TS can yield to better routing solution than those generated by the VRP illustrated in Figure 7.2. Let z(VRP) and z(VRP-TS) denote the optimal cost of a VRP and VRP-TS solutions, respectively. A fleet of homogeneous vehicles with capacity 5 is located at the depot, denoted by a square, and customers, represented by circles, have demands indicated by the numbers inside these circles. In the VRP solution, shown in Figure 7.2.b., three vehicles are utilized, each customer is visited with a back and forth trip from the depot, and the cost of this solution is z(VRP) = $6M$. Meanwhile, in the VRP-TS solution, Figure 7.2.c, two vehicles are used, where one vehicle, denoted by a dash line, visits the customer in the middle with a back and forth trip from the depot, and uses this location as a temporary storage in which 1 unit is left temporarily. The other vehicle, denoted by solid line, starts in the left corner of the figure, and then, it drives to pick up the additional 1

unit left at the temporary storage (its capacity increase from 2 to 3 units). Finally, this vehicle satisfies the demand of the unvisited customer. Thus, z(VRP-TS) = $4M + 2/M$. By letting M going to infinity, the ratio z(VRP)/z(VRP-TS) goes to 3/2.

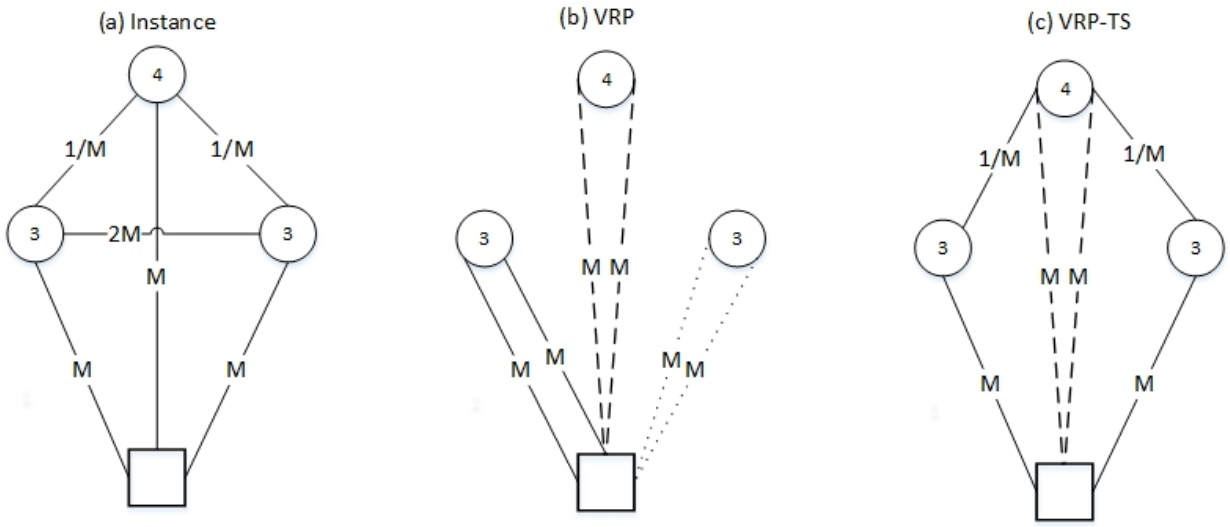

Figure 7.2: Example with unlimited number of vehicles: (a) problem instance, (b) vehicle routing problem (VRP) and (c) vehicle routing problem with temporary storage (VRP-TS).

VRP-TS might also improve the VRP-SD solutions, which is illustrated in Figure 7.3. In this example, we fix the number of vehicles to the minimum, which is calculated by $\lceil \sum_{i \in N'} d_i/C \rceil$, where N' denotes the set of customers to be visited with the demand of customers given by d_i and C represents the vehicle capacity. Thus, the minimum number of vehicles is $\lceil \sum_{i \in N'} d_i/C \rceil = (1 + 5 + 6 + 6 + 6)/8 = 3$. Let z(VRP-SD) denote the optimal cost of a VRP-SD solution. This solution, Figure 7.3.b, splits the demand of two customers, and its routing cost is z(VRP-SD) = $12M + 1/M$. On the other hand, in the VRP-TS solution, Figure 7.3.b, one vehicle (denoted by a dashed line) picks up 4 units left temporarily by the other two vehicles (denoted by solid and dotted lines) at the closest customer location from the depot, and its routing cost is z(VRP-TS) = $8M + 2/M$. By letting M going to infinity, the ratio z(VRP-SD)/z(VRP-TS) goes to 3/2.

An instance where z(VRP-SD) > z(VRP-TS), and assuming either a limited or an unlimited fleet size is presented in Figure 7.4. The vehicle capacity is 8 and the minimum number of vehicles, $\lceil \sum_{i \in N'} d_i/C \rceil = 2$. In the unlimited VRP-SD solution (equivalent to the VRP solution), Fig-

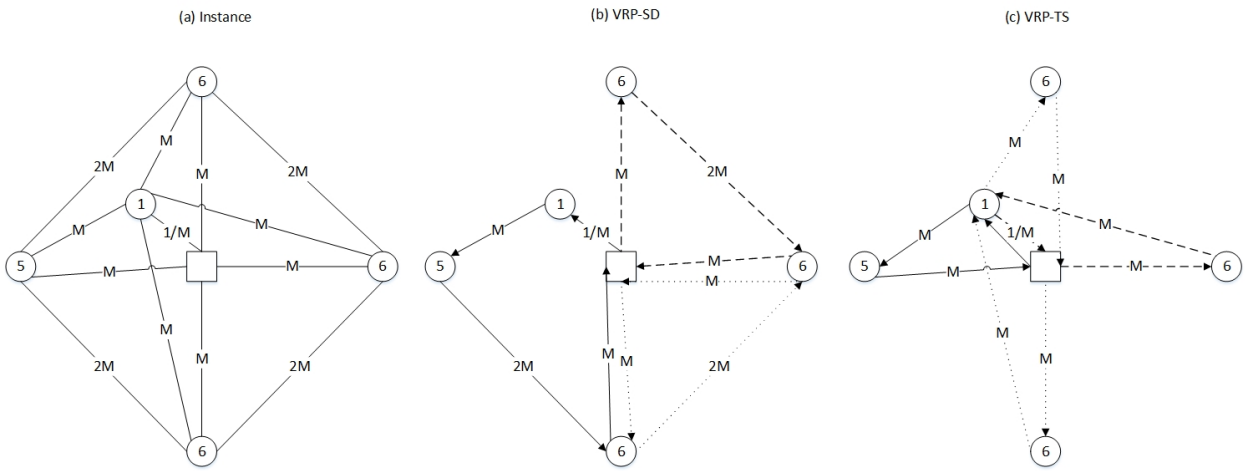

Figure 7.3: Example with minimum number of vehicles: (a) problem instance, (b) vehicle routing problem with split delivery (VRP-SD) and (c) vehicle routing problem with temporary storage (VRP-TS).

ure 7.4.(b), three vehicles are utilized, each customer is visited by a unique vehicle, and its routing cost is 393. In the limited fleet size VRP-SD solution (using minimum number of vehicles), Figure 7.4.(c), two vehicles are used, one customer is visited by two vehicles (denoted by solid and dotted lines), and its routing cost is 423. In the VRP-TS solution, Figure 7.4.(d), two vehicles are also utilized, but one customer is used as a temporary storage, where one vehicle (denoted by a solid line) drops 1 additional unit at this location. The other vehicle (denoted by a dash line) picks this extra unit at the temporary storage location, and delivers it to its final destination. The routing cost of this solution is 384. Therefore, VRP-TS might decrease both the number of vehicles and the routing cost compared with the unlimited fleet size VRP-SD solution.

Application areas where temporary storages can be used include customer locations owned by a company. Examples include biomass and humanitarian logistics, among others. This feature reduces both the number of vehicles and the routing cost.

We are also interested in combining both split delivery and temporary storage, and study their synergy. We denote this problem as VRP-SDTS. The relationship between VRP, VRP-SD, VRP-TS, and VRP-SDTS is illustrated in Figure 7.5. Note that all VRP solutions lie at the intersection of both VRP-SD and VRP-TS. Moreover, there are solutions in VRP-SD that are not in VRP-TS, and vice versa, and VRP-SDTS encompasses VRP, VRP-SD, and VRP-TS.

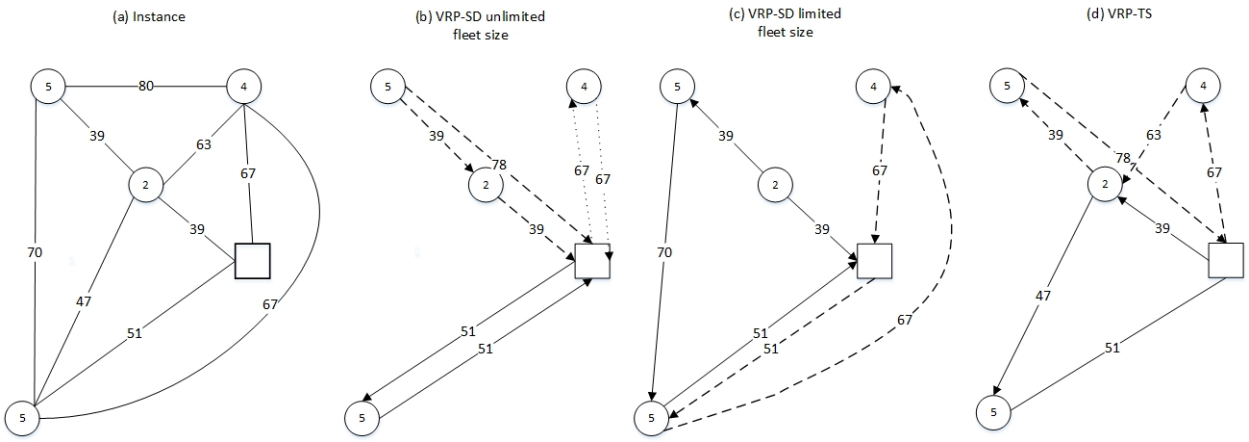

Figure 7.4: Example with unlimited and limited number of vehicles: (a) problem instance, (b) vehicle routing problem with split delivery (VRP-SD) and (c) vehicle routing problem with temporary storage (VRP-TS).

The remaining of this chapter is organized as follows. In Section 7.2, we present a review of related problems reported in the literature. In section 7.3, mathematical formulations for the VRP-TS and VRP-SDTS are introduced, and valid inequalities are derived. In Section 7.4, we develop properties of VRP-TS and VRP-SDTS, and finally, in Section 7.5, we present computational results on the performance of the proposed formulations for both VRP-TS and VRP-SDTS when solved directly by CPLEX.

7.2 Literature review

The idea of temporary storage is not new in the routing literature and has already been studied under several names including intermediate drops, preemption, reloads, transfers, or transshipment. However, all these designations have the same feature in common: vehicles transport items that can be dropped-off at intermediate points or transfer points; for pick up later by vehicles for delivery to their final destinations. Next, we present relevant papers that have used this concept.

Temporary storage has been studied under the name of intermediate drops. Atallah and Kosaraju (1988) studied a stacker crane problem, where a robot arm (unit capacity vehicle) has to rearrange different objects with known origin and destination in a line and a circle, and a object can be

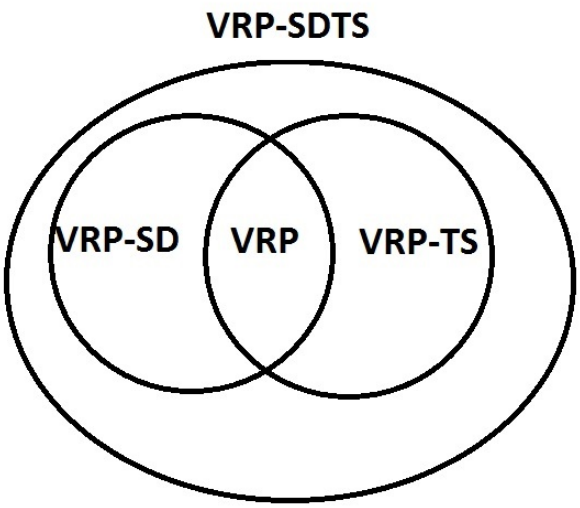

Figure 7.5: Relationship between VRP, VRP-SD, VRP-TS, and VRP-SDTS.

dropped-off before its final destination. Gribkovskaia and Laporte (2008) reviewed single vehicle pickup and delivery problems, where items do not have specified origin-destination, and a single vehicle located at the depot must make deliveries and pickups at customer locations. These authors only presented an example where the idea of intermediate drops is beneficial.

Temporary storage has also been presented under the name of preemption. Guan and Zhu (1998) studied a pickup and delivery problem on paths with preemption, where a capacitated single vehicle must satisfy all requests, with each having a known origin and a destination. Charikar et al. (2001) also studied a pickup and delivery problem with preemption, where a vehicle with capacity C must collect n unit that are distributed at arbitrary locations and deliver to n points each having a unit demand, and items can be dropped-off at intermediate locations. Bordenave et al. (2012) addressed the preemptive swapping problem (SP) in which several commodities without a specified origin-destination pair must be swapped between customer locations by a unique vehicle with capacity equal to 1, and commodities may be temporarily dropped-off at any vertex. Kerivin et al. (2012) presented a single-vehicle, and delivery problem with preemption, where a single vehicle with capacity C must transport a set of requests, each having a specified origin, destination, and volume. Salazar-González and Santos-Hernández (2015) studied the split-demand one-commodity pickup-and-delivery traveling salesman problem, where customers require or supply demand of a

single commodity, units of the commodity collected can satisfy any delivery location, and a single capacitated vehicle is located at the depot to satisfy all customer demands. In addition, customer demands can be split and locations can be used as temporary storage.

Temporary storage is also equivalent to reloads. Kerivin et al. (2008) studied a relaxation of the standard pickup and delivery problem in which each request has specified origin-destination, multiple capacitated vehicles are located at the depot, and split delivery and reloads are allowed. The problem consists of designing a set of routes for the vehicles so that all requests are delivered to their final destinations.

Temporary storage has also been reported as transfers. Shang and Cuff (1996) present a pickup and delivery problem with transfer motivated by a health-care application. Their problem considers uncapacitated vehicles, time windows, paired origin-destination for each request with ready time and due dates, and customers are allowed to transfer between vehicles at any location. Cortés et al. (2010) addressed an extension of the pickup and delivery problem in which passengers can transfer among vehicles at specific locations. Their problem consists of finding a set of routes for vehicles so that all transportation requests (passengers), each having known origin-destination pair, are transported to their final destinations. Masson et al. (2013) also presented a pickup and delivery problem with time windows in which requests can change vehicle during their trip at specific locations. They showed that transfer points can improve the value of the objective function by up to 9%. Masson et al. (2014) presented the dial-a-ride problem (DARP) with transfer and ride time constraints, where capacitated vehicles must be assigned to transportation request (users) that specify origins, destinations, and time windows. Furthermore, users might be transferred between vehicles at specific locations know as transfer points.

Temporary storage is equivalent to transshipments. Mitrovic-Minic and Laporte (2006) studied a pickup, delivery, and transshipment problem (PDP-T), where each request specified a origin-destination pair, transshipment is performed at specified point in the network, customers have time windows, and multiple uncapacitated vehicles are available. They concluded that transshipment can generate significant savings in routing cost when the requests are clustered. Qu and Bard (2012) and Rais et al. (2014) studied a pickup and delivery problem with transshipment, where

each request specify a origin, destination, and time window, and multiple heterogeneous vehicles are available.

Only Erdoğan et al. (2015) has used the term temporary storage in a pickup and delivery problem encountered in bicycle sharing systems, where stations to be visited require or supply bicycles, and a single capacitated vehicle is located at the depot to perform the rebalancing operations (picking up and delivering bicycles as necessary). In addition, customer demands can be split and locations can be used as temporary storage.

Table 7.1 presents a summary of the papers reviewed in this section, where we classify them into five categories. 1. Problem studied: the stacker crane problem (SCP), the pickup and delivery problems (PDP), the swapping problem (SP), and the Dial-a-Ride problem (DARP). 2. Transfer location: any or specific points in which a transfer occurs. 3. Commodities (products) in the problem: multiple or single. 4. Vehicles features. single with unit capacity, single with capacity C, multiple uncapacitated, or multiple with capacity C. 5. Origin-destination for each request: specified or not specified. All these previous studies have shown that allowing vehicles to transfer items among vehicles can significantly improve the solution value. However, none of the works presented has studied the traditional vehicle routing problem with temporary storage (VRP-TS). The VRP-TS is different from all different versions of pick up and delivery problems presented above for the following reasons. (i) The VRP-TS involves a single commodity, and all the items are initially located at the central depot. (ii) The items in the VRP-TS do not require predefined origin-destination pairs, i.e, any object can be deliver to any location.

Only Salazar-González and Santos-Hernández (2015) studied the split-demand one-commodity pickup-and-delivery traveling salesman problem and presented an approach that can be adapted to solve the VRP and VRP with split delivery and temporary storage (VRP-SDTS); nevertheless, they did not study in detail the impact of temporary storage itself, or show that temporary storage can yield savings identical to those by split delivery when compared with the VRP. We also differ from these authors by introducing a polynomial instead of an exponential-length formulation for the VRP-SD and VRP-SDTS.

Table 7.1: Classification of the papers involving the concept of transferring items between vehicles in the vehicle routing problem literature.

Reference	**1.Problem**	**2.Transfers point**	**3.Commodities**	**4.Vehicles**	**5.Origin-destination**
Atallah and Kosaraju (1988)	SCP	Any	Multiple	Single, C=1	Specified
Gribkovskaia and Laporte (2008)	PDP	Any	Multiple	Single, C=1	Not specified
Guan and Zhu (1998)	PDP	Any	Multiple	Single, C=1	Specified
Charikar et al. (2001)	PDP	Any	Multiple	Single, C=k	Specified
Bordenave et al. (2012)	SP	Any	Multiple	Single, C=1	Not specified
Kerivin et al. (2012)	PDP	Any	Multiple	Single, C=k	Specified
Salazar-González and Santos-Hernández (2015)	PDP	Any	Single	Single, C=k	Not specified
Kerivin et al. (2008)	PDP	Any	Multiple	Multiple, C=k	Specified
Shang and Cuff (1996)	PDP	Any	Multiple	Multiple, C=∞	Specified
Cortés et al. (2010)	DARP	Specific	Multiple	Multiple, C=k	Specific
Masson et al. (2013)	PDP	Specific	Multiple	Multiple, C=k	Specified
Masson et al. (2014)	DARP	Specific	Multiple	Multiple, C=k	Specified
Mitrovic-Minic and Laporte (2006)	PDP	Specific	Multiple	Multiple, C=∞	Specified
Qu and Bard (2012)	PDP	Specific	Multiple	Multiple, C=k	Specified
Rais et al. (2014)	PDP	Specific	Multiple	Multiple, C=k	Specified
Erdoğan et al. (2015)	PDP	Any	Single	Single, C=1	Not specified
Our research	**VRP**	**Any**	**Single**	**Multiple, C=k**	**Not specified**

7.3 Problem description, mathematical formulations, and valid inequalities

7.3.1 Problem description and mathematical formulations

The vehicle routing with split delivery and temporary storage (VRP-SDTS) can be defined as follows: *Given a complete graph with vertex set* $N = 1 \cup N'$, *where* 1 *denotes the depot in which a fleet of homogeneous vehicles with capacity* C *is located,* $N' = \{2, ..., n\}$ *comprises the set of customers with a demand of* d_i, $d_i \leq C$, $\forall i \in N'$, *and given a symmetric distance matrix* $[c_{ij}]$, *find a set of routes that satisfy the demand of all the customers while minimizing the total distance traveled by the vehicles, where each vehicle is assigned to a unique route, a customer demand can be satisfied by more than one vehicle, and a customer location can be used for temporary storage.*

The definition of the VRP-TS is identical to that of the VRP-SDTS except that each customer

demand is satisfied by one vehicle.

We present a mixed-integer programming formulation for the VRP-SDTS. We use the following notation.

Decision Variables:

- x_{ij}^f : binary variable, equals 1 if vehicle f traverses arc (i,j), and 0, otherwise. $\forall i,\ j \in N,\ j \neq i,\ \forall f \in \mathbb{F}$.
- y_{ij}^f : real variable that indicates the load on the vehicle f while traversing between customers i and j. $\forall i,\ j \in N,\ j \neq i,\ \forall f \in \mathbb{F}$.
- δ_i^f : real variable that indicates the amount delivered by vehicle f to customer i. $\forall i \in N',\ \forall f \in \mathbb{F}$.
- p_i^f : real variable that represents the amount picked up by vehicle f at customer location i. $\forall i \in N',\ \forall f \in \mathbb{F}$.

VRP-SDTS1 :

$$\text{Minimize } z = \sum_{f\in\mathbb{F}} \sum_{i\in N} \sum_{j\in N,\ j\neq i} c_{ij} x_{ij}^f \tag{7.1}$$

subject to :

$$\sum_{f\in\mathbb{F}} (\delta_i^f - p_i^f) = d_i \quad \forall i \in N' \tag{7.2}$$

$$\sum_{j\in N,\ j\neq 1} x_{1j}^f = 1, \quad \forall f \in \mathbb{F} \tag{7.3}$$

$$\sum_{j\in N,\ j\neq 1} x_{j1}^f = 1, \quad \forall f \in \mathbb{F} \tag{7.4}$$

$$\sum_{j\in N,\ j\neq i} x_{ij}^f - \sum_{j\in N,\ j\neq i} x_{ji}^f = 0, \quad \forall i \in N', \quad \forall f \in \mathbb{F} \tag{7.5}$$

$$y_{ij}^f \leq C x_{ij}^f, \quad \forall i,j \in N, \quad j \neq i, \quad \forall f \in \mathbb{F} \tag{7.6}$$

$$\sum_{j\in N,\ j\neq i} y_{ji}^f - \sum_{j\in N,\ j\neq i} y_{ij}^f = \delta_i^f - p_i^f, \quad \forall i \in N', \quad \forall f \in \mathbb{F} \tag{7.7}$$

$$\delta_i^f \leq C \sum_{j\in N,\ j\neq i} x_{ji}^f, \quad i \in N', \quad \forall f \in \mathbb{F} \tag{7.8}$$

$$p_i^f \leq C \sum_{j\in N,\ j\neq i} x_{ji}^f, \quad i \in N', \quad \forall f \in \mathbb{F} \tag{7.9}$$

$$x_{ij}^f \in \{0,1\}, \quad \forall i,j \in N, \quad j \neq i, \quad \forall f \in \mathbb{F} \tag{7.10}$$

$$y_{ij}^f \geq 0, \quad \forall i,j \in N, \quad j \neq i, \quad \forall f \in \mathbb{F} \tag{7.11}$$

$$p_i^f \geq 0, \quad \forall i \in N', \quad \forall f \in \mathbb{F} \tag{7.12}$$

$$\delta_i^f \geq 0, \quad \forall i \in N', \quad \forall f \in \mathbb{F}. \tag{7.13}$$

Constraints (7.2) enforce that the balance between amounts delivered and picked up for all vehicles at location i is equal to its demand. Constraints (7.3) and (7.4) make sure that each vehicle departs from and returns to the depot. Constraints (7.5) are the standard flow balance constraints. Constraints (7.6) and (7.7) are the subtour elimination constraints. Constraints (7.6) enforce that the flow of each arc cannot exceed the given vehicle capacity, and Constraints (7.7) determine the amount delivered and picked up at location i for vehicle f. Constraints (7.8) and (7.9) enforce that vehicle f can deliver and pick up from location i only if f has visited this location. Constraints (7.10)- (7.13) are the domains of the decision variables. By letting $p_i^f = 0, \ \forall i \in N, \ \forall f \in \mathbb{F}$, the VRP-SDTS formulation reduces to that for the VRP-SD, i.e, vehicles are only allowed to drop-off, but not pick-up items. To model the VRP-TS, we need an additional binary variable, z_i^f, $\forall i \in N'$, $\forall f \in \mathbb{F}$, that indicates if vehicle f delivers the demand and possibly drops-off additional units temporarily at location i. Furthermore, we need to add to the VRP-SDTS1 formulation the following constraints:

$$\sum_{j \in N, \ j \neq i} x_{ji}^f \leq z_i^f, \quad i \in N', \quad \forall f \in \mathbb{F} \tag{7.14}$$

$$\sum_{f \in \mathbb{F}} z_i^f = 1, \quad \forall i \in N' \tag{7.15}$$

$$\delta_i^f \geq d_i z_i^f, \quad \forall i \in N', \quad \forall f \in \mathbb{F}. \tag{7.16}$$

Constraints (7.14) and (7.15) capture if vehicle f visits customer i and delivers its demand, respectively. Finally, Constraints (7.16) impose that a unique vehicle must satisfy the demand of location i, which prohibits split delivery solutions.

7.3.2 A two-index formulation for the VRP-SDTS

We present an alternative representation of the VRP-SDTS, a two-index flow-based formulation denoted by VRP-SDTS2, and then we prove that VRP-SDTS1 and VRP-SDTS2 are equivalent. We use the following additional decision variables to introduce VRP-SDTS2: x_{ij} a integer variable that indicates the number of vehicles traveling from city i to city j, $\forall i, j \in N,\ j \neq i$, and y_{ij} a real variable that indicates the load of the vehicle when traveling from city i to city j, $\forall i, j \in N,\ j \neq i$.

VRP-SDTS2 :

$$\text{Minimize} \quad z = \sum_{i \in N} \sum_{j \in N,\ j \neq i} x_{ij} \tag{7.17}$$

$$\sum_{j \in N,\ j \neq 1} x_{1j} = \mathbb{F} \tag{7.18}$$

$$\sum_{j \in N,\ j \neq 1} x_{j1} = \mathbb{F} \tag{7.19}$$

$$\sum_{j \in N,\ j \neq i} x_{ij} - \sum_{j \in N,\ j \neq i} x_{ji} = 0, \quad \forall i \in N' \tag{7.20}$$

$$y_{ij} \leq C x_{ij}, \quad \forall i, j \in N,\ j \neq i \tag{7.21}$$

$$\sum_{j \in N,\ j \neq i} y_{ji} - \sum_{j \in N,\ j \neq i} y_{ij} = d_i, \quad \forall i \in N' \tag{7.22}$$

$$x_{ij} \in Z^{+} \quad \forall i, j \in N,\ j \neq i \tag{7.23}$$

$$y_{ij} \geq 0 \quad \forall i, j \in N,\ j \neq i. \tag{7.24}$$

Constraints (7.18) and (7.19) enforce that exactly F vehicles depart from and return to the depot. Constraints (7.20) are the standard flow balance constrains. Constraints (7.21) and (7.22) are the subtour eliminations constraints. Constraints (7.21) enforce that the flow of each arc cannot exceed the vehicle capacity times the number of vehicles traversing arc (i, j), and Constraints (7.22) make sure that the balance of the flow-out and flow-in for each customer is equal to its demand d_i. Constraints (7.23) and (7.24) are the domains of the decision variables.

Proposition 7.3.1. *Formulations VRP-SDTS1 and VRP-SDTS2 are equivalent.*

Proof. We show that for each feasible solution to formulation VRP-SDTS1 there exists a feasible solution to formulation VRP-SDTS2, and vice versa.

case (a). $(x^f, y^f, p^f, \delta^f) \implies (x, y)$

Let $(x^f, y^f, p^f, \delta^f)$ be a feasible solution of VRP-SDTS1. Let $x_{ij} = \sum_{f\in\mathbb{F}} x_{ij}^f$ and $y_{ij} = \sum_{f\in\mathbb{F}} y_{ij}^f$, $\forall i, j \in N$, $i \neq j$. Then, x_{ij} satisfies (7.18) and (7.19), and will also satisfy the flow conservation constraints, (7.20). Moreover, y_{ij} represents the total load transported by vehicles between customers i and j. If y_{ij}^f satisfies (7.6), $\forall i, j \in N$, $j \neq i$, $\forall f \in \mathbb{F}$, then, they clearly satisfy (7.21). Moreover, (7.2) and (7.7)- (7.9) imply (7.22). Therefore, x_{ij}, y_{ij}, $\forall i, j \in N$, $j \neq i$, is a feasible solution of VRP-SDTS2.

case (b) $(x, y) \implies (x^f, y^f, p^f, \delta^f)$

Let x_{ij}, y_{ij}, $\forall i, j \in N$, $j \neq i$, be a feasible solution of VRP-SDTS2. Consider the following:

1. For each arc (i, j) $\forall i, j \in N$, $j \neq i$, create as many arcs as the numerical value of x_{ij}, and label each arc as $(i, j)_u$, where $u = 1, ..., x_{ij}$.
2. For each vehicle f, compute x_{ij}^f by performing the following operations. Let $f = 1$.
   (a) Starting from the depot, select a unmarked arc $(1, i)_u$, for an arbitrary i and u, then, assign vehicle f to arc $(1, i)_u$, i.e, $x_{1i}^f = 1$, and delete arc $(1, i)^u$. If no arc $(1, i)^u$ is available, then stop and go to 3; otherwise, go to (b).
   (b) Select an existing and unmarked out coming arc from i $(i, j)^u$ for some $j \in N$, and $u \in x_{ij}$. Assign vehicle f to arc $(1, i)_u$, i.e, $x_{ij}^f = 1$, and delete arc $(i, j)^u$ as selected. If $j = 1$, then a complete route has been created, increase f by 1 and go to 1. Otherwise, make $i = j$ and repeat (b).
3. To compute δ_i^f, p_i^f, and y_{ij}^f do the following
   (a) Make all variables $x_{ij}^f = 1$ for the arcs selected in step 1, and $x_{ij}^f = 0$ for the arcs not selected in Step 1. Then, solve the network flow model given by Constraints (7.2), and (7.6)- (7.9) once all x_{ij}^f values have been specified, and obtain δ_i^f, p_i^f, and y_{ij}^f.

If x_{1i}, $\forall i \in N$, satisfies (7.18) and (7.19), then x_{1i}^f will satisfy (7.3) and (7.4) given that F routes that start and end at the depot have been constructed. Furthermore, x_{ij}^f, $\forall i, j \in N$, $j \neq i$, $\forall f \in \mathbb{F}$,

will satisfy the flow conservation constrains (7.5) for each vehicle given that for each arc (i, j) in a route, there is a unique out-coming arc from (j, k), $\forall(i, j)$ and $\forall(j, k) \in A$. Obviously, δ_i^f, p_i^f, and y_{ij}^f will satisfy Constraints (7.2), and (7.6)- (7.9) given that this system of equations was solved to obtain its values. Therefore, x_{ij}^f, y_{ij}^f, δ_i^f, and p_i^f, $\forall i, j \in N,\ j \neq i,\ \forall f \in \mathbb{F}$, is a feasible solution of VRP-SDTS1.

□

7.3.3 A two-index formulation for the VRP-TS

We also propose a two-index single commodity flow formulation for the VRP-TS by copying each city i, $i \in N-\{1\}$, exactly once, thus, i is replaced by i_1 and i_2, where the former node, i_1, represents a demand point requiring d_i units, and i_2 denotes a temporary storage requiring a demand of 0 unit. At i_2 locations, items are transferred among vehicles. We define three new sets as follows: $N' = i_1$ and $S' = i_2$, $\forall i \in N - \{1\}$, and $NS' = N' \cup S'$. Therefore, we have the new node set $\hat{N} = \{1 \cup NS'\}$. Furthermore, letting z_i, $i \in \hat{N} - \{1\}$, be a variable representing the number of times city i is visited, the VRP-TS can be formulated as follows:

VRP-TS :

$$\text{Minimize} \quad z = \sum_{i \in \hat{N}} \sum_{j \in \hat{N},\ j \neq i} x_{ij} \tag{7.25}$$

$$\sum_{j \in \hat{N},\ j \neq 1} x_{1j} = \mathbb{F} \tag{7.26}$$

$$\sum_{j \in \hat{N},\ j \neq 1} x_{j1} = \mathbb{F} \tag{7.27}$$

$$\sum_{j \in \hat{N},\ j \neq i} x_{ij} = z_i, \quad \forall i \in NS' \tag{7.28}$$

$$\sum_{j \in \hat{N},\ j \neq i} x_{ji} = z_i, \quad \forall i \in NS' \tag{7.29}$$

$$z_i = 1, \quad \forall i \in N' \tag{7.30}$$

$$\sum_{j \in \hat{N},\ j \neq i} y_{ji} - \sum_{j \in \hat{N},\ j \neq i} y_{ij} = d_i, \quad \forall i \in NS' \tag{7.31}$$

$$y_{ij} \leq C x_{ij}, \quad \forall i, j \in \hat{N},\ j \neq i \tag{7.32}$$

$$y_{ij} \geq 0 \quad \forall i, j \in \hat{N},\ j \neq i \tag{7.33}$$

$$z_i \in Z^+ \quad \forall i \in NS' \tag{7.34}$$

$$x_{ij} \in Z^+ \quad \forall i, j \in \hat{N},\ j \neq i \tag{7.35}$$

$$\tag{7.36}$$

Proposition 7.3.2. *The following inequalities are valid for the VRP-TS.*

$$x_{ij} \in \{0,1\}, \quad \forall i, j \in N', \quad i \neq j \tag{7.37}$$

$$x_{ij} \in \{0,1\}, \quad \forall i \in N',\ \forall j \in S', \quad i \neq j \tag{7.38}$$

$$x_{ij} + x_{ji} \leq 1, \quad \forall i, j \in N', \quad i \neq j, \quad i < j \tag{7.39}$$

$$x_{ij} + x_{ji} \leq 1, \quad \forall \in N', \quad \forall j \in S, \quad i \neq j, \quad i < j \tag{7.40}$$

Proof. Constraints (7.37)-(7.40) follow from Constraints (7.30) □

7.4 Properties of VRP-TS and VRP-SDTS

7.4.1 Properties of the VRP-TS

Proposition 7.4.1. *VRP-TS is an NP-Hard problem.*

Proof. By letting $z_i = 0$, $\forall i \in S'$, the VRP-TS reduces to the VRP problem, which have been shown to be NP-Hard. □

Proposition 7.4.2. *z(VRP-TS) ≤ z(VRP)*

Proof. By letting $z_i = 0, \forall i \in S'$, the VRP-TS formulation reduces to that for the VRP. Therefore, every VRP solution is in VRP-TS, and it follows that c(VRP-TS) $\not>$ c(VRP). To show that c(VRP-TS) $\leq$ c(VRP), it is sufficient to show an example, where c(VRP-TS) $<$ c(VRP), which in shown in Figure 7.2. □

Proposition 7.4.3. *There always exists a solution to VRP-TS with the minimum number of vehicles ($\lceil \sum_{i \in N'} d_i / C \rceil$).*

Proof. We prove this by creating a solution to the VRP-TS with the minimum number of vehicles. Let $m = \lceil \sum_{i \in N'} d_i / C \rceil$, and for routes $r = 1, ..., m-1$, we perform the following steps to add customers to each of these routes. Let $r = 1$.

1. Add an unvisited customer location i to route r, i.e, $r = \{r \cup i\}$. If $d_i < C$, then $C = C - d_i$ and mark i as visited, and go to (2).

2. If there exits an unvisited customer location i with $d_i < C$, then add i to r, let $C = C - d_i$, and repeat (2); otherwise, go to (3).

3. Drop $d_i - C$ units at location i and add i to the set of temporary storage denoted by $T_s = \{T_s \cup i\}$, increase r by one, and go to (1).

The last route, $r = m$ is created as follows.

1. Add an unvisited customer location i to route r, i.e, $r = \{r \cup i\}$. If $d_i < C$, then $C = C - d_i$ and mark i as visited. If there is no more unvisited customers, then stop; otherwise, go to (2).

2. If there exits an unvisited customer location i with $d_i < C$, then add i to r, let $C = C - d_i$, and repeat (2); otherwise, go to (3).

3. Travel from location i to a location s in T_s to pick the items dropped so that the capacity of this route has increased to C or there are no more items available to pick up. If there are no more items available at location $s \in T_S$, then delete s from T_s. Finally, go to (1).

Since $c\lceil \sum_{i \in N'} d_i / C \rceil > \sum_{i \in N'}$ all demands will be satisfied, and therefore, we have created a feasible solution to VRP-TS using the minimum number of vehicles.

□

Let z(VRP) and z(VRP-TS) denote the cost of the VRP and VRP-TS optimal solutions, respectively. Archetti et al. (2006) proved that z(VRP)/z(VRP-SD) ≤ 2, and the bound is tight, i.e, there is an example where this bound holds. However, there is to known bound for z(VRP)/z(VRP-TS) at this time. We can make the following observation.

Observation 7.4.1. *There exists an example in which* $\frac{z(VRP)}{z(VRP\text{-}TS)} = 2$

To show this, we use an instance similar to the one presented in Archetti et al. (2006), where the first set of n customers are located at distance 1 from the depot, and additional n customers located at distance $1 + 1/M$ from the depot perfectly aligned with the first set. The distance between any pair of customers is $1/M$. Each customer has a demand of $C/2 + 1$, and assume $C \geq 2n$. An optimal VRP solution consists of back and forth trips visiting each customer, thus, its routing cost is $4n + 2n/M$. On the other hand, the VRP-TS solution consists of $n + 1$ routes. The first n routes deliver $C/2+1$ units to satisfy the demand of the closest n customers, and they drop-off the remaining amount at each location, i.e, $C - (C/2 + 1) = C/2 - 1$ which will be picked up later for the last route denoted by $n + 1$. Route $n + 1$ visits a customer at distance $M + 1/M$ and delivers $C/2+1$ (its capacity reduces to $C/2-1$). Note that $n(C/2-1)$ units are available to pick up at the n temporary locations so the total units still available to deliver are $(n+1)(C/2-1)$. Meanwhile, demand to be satisfied is $(n-1)(C/2+1)$. First, we show that the total units available are greater than the demand to be satisfied, i.e, $(n+1)(C/2-1) - (n-1)(C/2+1) = C - 2n \geq 0$. The cost of the VRP-TS consists of n routes of length 2, and the additional route of length $2 + (n-1)1/M$. Thus, $\frac{\text{z(VRP)}}{\text{z(VRP-TS)}} = \frac{4n+2n/M}{2n+(n-1)1/M+2} = \frac{4+2/M}{2+2/nM+(n-1)/nM+2/n}$. By letting $lim_{M\to\infty}$, and $lim_{n\to\infty}$, it follows that $\frac{\text{z(VRP)}}{\text{z(VRP-TS)}} = 2$. Thus, the results follows.

7.4.2 Properties of the VRP-SDTS

Proposition 7.4.4. *VRP-SDTS is an NP-Hard problem.*

Proof. By fixing $p_i^f = 0$, $\forall i \in N$, $\forall f \in \mathbb{F}$, the VRP-SDTS reduces to that for VRP-SD, which have been shown to be NP-Hard. □

Proposition 7.4.5. *z(VRP-SDTS) ≤ z(VRP-SD) and z(VRP-SDTS) ≤ z(VRP-TS)*

Proof. By fixing $p_i^f = 0$, $\forall i \in N$, $\forall f \in \mathbb{F}$, the VRP-SDTS formulation reduces to that for VRP-SD. Therefore, every VRP-SD solution is in VRP-SDTS, and it follows that c(VRP-SDTS) $\not>$ c(VRP-SD). To show that c(VRP-SDTS) $\leq$ c(VRP-SD), it is sufficient to show an example, where c(VRP-SDTS) $<$ c(VRP-SD), which in shown in Figure 7.4. Similarly, we can show z(VRP-SDTS) $\leq$ z(VRP-TS). □

Proposition 7.4.6. *There always exists a solution to VRP-SDTS with the minimum number of vehicles ($\lceil \sum_{i \in N'} d_i / C \rceil$).*

Proof. Similar to 7.4.3 □

Proposition 7.4.7. *At a given location i, a vehicle f can only deliver or pick up items, but not both.*

Proof. Let define $d_i^f = \delta_i^f - p_i^f$ be the amount of demand at location i satisfied by vehicle f, where δ_i^f indicates the amount delivered, and p_i^f represents the amount picked up at location i. Furthermore, let assume route f visit customer k first, then i, and afterwards j. Thus, $y_{ij}^f = y_{ki}^f - \delta_i^f + p_i^f$ represents the remaining load after visiting customer i,

- case (a) $d_i^f = 0$.

  If $d_i^f = 0$, then $\delta_i^f = p_i^f$, which implies that vehicle f delivers and picks up the same quantity, by the triangle inequality deleting customer i from route f can only decrease the total routing cost, and thus the load remaining on route f is unchanged.

- case (b) $d_i^f > 0$.

  If $d_i^f > 0$, we have that $\delta^f > p_i^f$. Then, a new feasible solution can be constructed such that either δ^f or p_i^f is zero. To do so, we define the following new variables, $\hat{\delta}^f = \delta_i^f - p_i^f$, and $\hat{p}_i^f = 0$. Then, $\hat{y_{ij}^f} = y_{ki}^f - \hat{\delta}^f + \hat{p}^f = y_{ki}^f - \delta_i^f + p_i^f$, and thus the load remaining on route f is unchanged.

- case (c) $d_i^f < 0$

  If $d_i^f < 0$, we have $\delta^f < p_i^f$, and as before a new feasible solution can be constructed, where $\hat{\delta}^f = 0$. $\hat{\delta}^f = 0$, and $\hat{p}_i^f = p_i^f - \delta_i^f$. It follows that $\hat{y}_{ij}^f = y_{ki}^f - \hat{\delta}^f + \hat{p}^f = y_{ki}^f - \delta_i^f + p_i^f$, and therefore the load remaining of route f is unchanged.

□

Let z(VRP-SDTS) denote the cost of a VRP-SDTS optimal solution, respectively. We can make the following observation.

Observation 7.4.2. *There exists an example in which* $\frac{z(VRP)}{z(VRP\text{-}SDTS)} = 2$

We can show this by noticing that the solution described in Observation 7.4.1 is a feasible solution to VRP-SDTS.

7.5 Computational Results

In this section, we present computational results obtained by solving the two-index formulations proposed for the VRP-TS and VRP-SDTS directly by the commercial solver CPLEX. The runs were made using CPLEX 12.5.1 with OPL 6.1 on a computer having 32 GB of RAM and running windows 7 with a maximum time limit of one hour of wall clock time. To test our formulations, we have used three well-known benchmark instances available in the literature for the VRP-SD. *Set 1* consists of 11 instances derived from TSPLIB (1997), where the number of customers varies from 50 to 100. *Set 2* contains 14 randomly generated instances proposed by Belenguer et al. (2000), where the coordinates correspond to instances taken from the TSPLIB, and the vehicle capacity is 100. Finally, *Set 3* includes 12 instances proposed by Chen et al. (2007), where the number of customers varies from 8 to 288, the vehicle capacity is 100, and the customers are located in concentric circles around the depot each having a demand of 60 or 90.

In all instances, the number of vehicles was set to the minimum, $\mathbb{F} = \lceil \sum_{i \in N'} d_i / C \rceil$, and our results are compared with the best known solutions reported for the VRP-SD in Archetti et al. (2014).

7.5.1 Results for VRP-TS

In Table 7.2, we present results obtained by solving the two-index formulation proposed for VRP-TS directly by CPLEX on instances from Set 1. Columns 1-3 indicate, respectively, the problem name (Name), the number of customers (n), and the best known solution reported for the VRP-SD in the literature (SD). Columns 4-6 present, respectively, the best lower bound (Z_{LB}), the integer solution (Z_{IP}), and the CPU time in seconds to reach optimality or the integer optimality gap obtained by CPLEX for the solution of VRP-TS. The symbols tl denote that a feasible solution could not be obtained for that instance in the prescribed time limit. If a feasible solution was obtained within the time limit, then we report the gap value. The last column (% from SD) denote the percentage from the SD solution. On average, VRP-TS obtained solutions with a routing cost that are 2.3% above the best known VRP-SD solutions. However, we remark that for one instance, eil30, VRP-TS was able to generate a better solution than the optimal solution for the VRP-SD. We have highlighted in bold (in this and the remaining tables) instances in which VRP-TS obtained solutions with similar or smaller cost than those reported for VRP-SD in the literature.

Table 7.2: Results obtained by solving the two-index formulation proposed for VRP-TS on instances from *Set 1*.

Name	*n*	*SD*	Z_{LB}	Z_{IP}	*T/gap*	*% from SD*
eil22	21	375	353.59	375	5.71%	**0.0%**
eik23	22	569	569	569	143.27	**0.0%**
eil30	29	510	438.74	508	13.63%	**-0.4%**
eil33	32	835	785.66	835	5.91%	**0.0%**
eil51	50	521	500.22	523	4.36%	0.4%
eilA76	75	828	749.68	855	12.32%	3.2%
eilB76	75	1019	895.46	1096	18.30%	7.0%
eilC76	75	735	661.57	773	14.42%	4.9%
eilD76	75	682	613.18	705	13.02%	3.3%
eilA101	100	817	776.1	854	9.12%	4.3%
eilB101	100	1077	991.9	1147	13.52%	6.1%

Table 7.3 presentS results obtained by solving the two-index formulation proposed for VRP-TS directly by CPLEX on instances from Set 2. The columns are identical to those presented in previous table. On average, CPLEX obtained solutions with costs that were 5.9% above the best

known VRP-SD solutions. We observe that as the number of customers increases, the problem become more difficult to solve.

Table 7.3: Results obtained by solving the two-index formulation proposed for VRP-TS on instances from *Set 2*.

Name	*n*	*SD*	Z_{LB}	Z_{IP}	*T/gap*	*% from SD*
S51D1	50	458	452.47	458	1.21%	**0.0%**
S51D2	50	703	636.57	708	10.09%	0.7%
S51D3	50	944	860.36	950.00	9.44%	0.6%
S51D4	50	1551	1444.9	1646	12.22%	5.8%
S51D5	50	1328	1241.34	1397	11.14%	4.9%
S51D6	50	2221	2032.01	2261	10.13%	1.77%
S76D1	75	592	575.32	619	7.06%	4.4%
S76D2	75	1097	963.33	1225	21.36%	10.4%
S76D3	75	1455	1282.95	1650	22.25%	11.8%
S76D4	75	2111	1906.53	2259	15.60%	6.6%
S101D1	100	716	621.93	883	29.57%	18.9%
S101D2	100	1391	1198.99	1396	14.11%	0.4%
S101D3	100	1916	1677.83	2023	17.06%	5.3%
S101D5	100	2918	2512.37	3278	23.36%	11.0%

Table 7.4 present results obtained by solving the two-index formulation proposed for VRP-TS directly by CPLEX on 12 instances from Set 3. Columns are identical to those presented in previous table. CPLEX was able to obtain feasible solutions to 10 out of the 12 cases, and, for these instances, the routing cost was 0.3% above the best known VRP-SD solutions.

7.5.2 Results for VRP-SDTS

For each solution of the VRP-SDTS, we perform an analysis to identify the type of deliveries performed, where RD, SD, TS, denote, respectively, a regular, a split, and a temporary delivery (temporary storage). We use $RD + SD$ to denote that a solution contains both RD and SD. $RD + TS$ and $RD + SD + TS$ are defined similarly.

Table 7.5 displays results obtained by solving the two-index formulation proposed for VRP-SDTS directly by CPLEX on instances from Set 1. Columns 1-6 are identical to those presented in previous

Table 7.4: Results obtained by solving the two-index formulation proposed for VRP-TS on instances from *Set 3*.

Name	*n*	*SD*	Z_{LB}	Z_{IP}	*T/gap*	*% from SD*
SD1	8	228	228	228	4.26	**0.0%**
SD2	16	708	708	708	131.68	**0.0%**
SD3	16	432	432	432	105.22	**0.0%**
SD4	24	630	630	630	2325.38	**0.0%**
SD5	32	1392	1328.99	1392	4.53%	**0.0%**
SD6	32	832	815.22	832	2.27%	**0.0%**
SD7	40	3640	3368.28	3640	7.46%	**0.0%**
SD8	48	5068	4706.73	5068	7.13%	**0.0%**
SD9	48	2046	1947.7	2061	5.50%	**0.7%**
SD10	64	2688	2496.44	2741	8.92%	**1.9%**
SD11	80	13280	12697	-	tl	-
SD12	80	7315	6684	-	tl	-

tables, and Column 7-10 indicate the type of delivers used in each solution. On average, VRP-SDTS obtained solutions that were 0.76% above the cost of the best known VRP-SD solutions. Note that for instance eil30, VRP-SDTS generates a solution that has a lower routing cost than the optimal VRP-SD solution. With respect to the type of deliveries performed, three solutions present regular deliveries (VRP solutions), two solutions each contain regular and temporary deliveries (VRP-TS solutions), and five solutions perform regular, split, and temporary deliveries (VRP-SDTS solutions). Note that CPLEX obtained the same or smaller cost for VRP-SDTS than that for the VRP-SD solutions in 6 out of the 11 instances.

In Table 7.6, we report results obtained by solving the two-index formulation proposed for VRP-SDTS directly by CPLEX on instances from Set 2. Columns are identical to those displayed in previous table. On average, VRP-SDTS obtained solutions that were 1.90% above the cost of VRP-SD solutions. Moreover, for instance S51D6, the VRP-SDTS solution had a lower cost than that for VRP-SD. With respect to the type of deliveries executed, only one solution involves regular deliveries (VRP solution), one solution performs both regular and split deliveries (VRP-SD solution), two solutions contain regular and temporary deliveries (VRP-TS solutions), and ten solutions perform regular, split, and temporary deliveries (VRP-SDTS solutions).

Table 7.5: Results obtained by solving the two-index formulation proposed for VRP-SDTS on instances from *Set 1*.

Name	*n*	*SD*	Z_{IP}	*T/gap*	*% from SD*	*RD*	*RD+SD*	*RD+TS*	*RD+SD+TS*
eil22	21	375	375	12.34	**0.00%**	x			
eik23	22	569	569	1.01	**0.00%**	x			
eil30	29	510	508	5.34	**-0.39%**			x	
eil33	32	835	835	1.84%	**0.00%**			x	
eil51	50	521	521	1.21%	**0.00%**	x			
eilA76	75	828	841	7.82%	1.55%				x
eilB76	75	1019	1028	9.11%	0.88%				x
eilC76	75	735	735	4.70%	**0.00%**			x	
eilD76	75	682	685	4.68%	0.44%				x
eilA101	100	817	823	5.55%	0.73%				x
eilB101	100	1077	1136	12.11%	5.19%				x

Table 7.6: Results obtained by solving the two-index formulation proposed for VRP-SDTS on instances from *Set 2*.

Name	*n*	*SD*	Z_{IP}	*T/gap*	*% from SD*	*RD*	*RD+SD*	*RD+TS*	*RD+SD+TS*
S51D1	50	458	458	45.52	**0.00%**	x			
S51D2	50	703	708	4.28%	0.71%				x
S51D3	50	944	947	3.93%	0.32%				x
S51D4	50	1551	1565	4.39%	0.89%				x
S51D5	50	1328	1345	4.44%	1.26%				x
S51D6	50	2221	2158	2.26%	**-2.92%**		x		
S76D1	75	592	595	1.16%	0.50%			x	
S76D2	75	1097	1106	7.77%	0.81%				x
S76D3	75	1455	1455	6.73%	0.00%				x
S76D4	75	2111	2165	8.31%	2.49%				x
S101D1	100	716	732	4.86%	2.19%			x	
S101D2	100	1391	1552	17.13%	10.37%				x
S101D3	100	1916	2097	15.70%	8.63%				x
S101D5	100	2918	2957	10.27%	1.32%				x

Table 7.7: Results obtained by solving the two-index formulation proposed for VRP-SDTS on instances from *Set 1* with a time limit of two hours of wall clock time.

Name	*n*	*SD*	Z_{IP}	*T/gap*	*% from SD*
eil22	21	375	375	7.97	**0.0%**
eik23	22	569	569	0.86	**0.0%**
eil30	29	510	510	9.84	**0.0%**
eil33	32	835	835	12.14	**0.0%**
eil51	50	521	521	0.52%	**0.0%**
eilA76	75	828	823	1.64%	**-0.6%**
eilB76	75	1019	1015	3.00%	**-0.4%**
eilC76	75	735	733	1.30%	**-0.3%**
eilD76	75	682	684	1.71%	0.3%
eilA101	100	817	818	1.79%	0.1%
eilB101	100	1077	1066	1.00%	**-1.0%**

Impact on solving VRP-SDTS with a time limit of two hours

We also investigate the effects of increasing the allowed time limit to two hours instead of one hour of wall clock time. Table 7.7 displays results obtained on instances from Set 1. VRP-SDTS obtained the same or smaller cost than that for VRP-SD solutions in 9 out the 11 instances, and note that for instances eilA76, eilB76, eilC76, and eilB101, we obtained better solutions than that for VRP-SD. The largest improvement occurred for eilB101, where the cost was reduced by 1.0%.

Table 7.8 presents results obtained on instances from Set 2 when the time limit is increased to two hours. VRP-SDTS obtained the same or smaller cost than that for VRP-SD solutions in 10 out the 14 instances, and note that for instances SD51D6, SD76D2, SD76D2, and SD101D5, we obtained better solutions than that for VRP-SD. The largest improvement occurred for SD101D5, where the cost was reduced by 3.8%.

Table 7.9 presents results obtained on instances from Set 3 when the time limit is increased to two hours. VRP-SDTS obtained the same or smaller cost than that for VRP-SD solutions in 13 out of the 13 instances, and note that for instance SD13, we obtained a feasible solution for which there is no known solution available for the VRP-SD.

Table 7.8: Results obtained by solving the two-index formulation proposed for VRP-SDTS on instances from *Set 2* with a time limit of two hours of wall clock time.

Name	*n*	*SD*	Z_{IP}	*T/gap*	*% from SD*
S51D1	50	458	458	25.16	**0.0%**
S51D2	50	703	703	0.80%	**0.0%**
S51D3	50	944	944	56.24	**0.0%**
S51D4	50	1551	1551	0.35%	**0.0%**
S51D5	50	1328	1328	0.71%	**0.0%**
S51D6	50	2221	2163	0.70%	**-2.7%**
S76D1	75	592	592	0.23%	**0.0%**
S76D2	75	1097	1093	2.45%	**-0.4%**
S76D3	75	1455	1440	2.31%	**-1.0%**
S76D4	75	2111	2114	2.93%	0.1%
S101D1	100	716	720	1.02%	0.6%
S101D2	100	1391	1397	3.00%	0.4%
S101D3	100	1916	1925	4.22%	0.5%
S101D5	100	2918	2811	1.98%	**-3.8%**

Table 7.9: Results obtained by solving the two-index formulation proposed for VRP-SDTS on instances from *Set 3* with a time limit of two hours of wall clock time.

Name	*n*	*SD*	Z_{IP}	*T/gap*	*% from SD*
SD1	8	228	228	0.59	**0.0%**
SD2	16	708	708	0.95	**0.0%**
SD3	16	432	432	0.01	**0.0%**
SD4	24	630	630	2.40	**0.0%**
SD5	32	1392	1392	2199.74	**0.0%**
SD6	32	832	832	180.13	**0.0%**
SD7	40	3640	3640	0.03%	**0.0%**
SD8	48	5068	5068	0.00%	**0.0%**
SD9	48	2046	2046	0.64%	**0.0%**
SD10	64	2688	2688	0.21%	**0.0%**
SD11	80	13280	13280	1.10%	**0.0%**
SD12	80	7315	7315	1.93%	**0.0%**
SD13	96	-	10110	1.33%	-

7.5.3 Discussion

Given the results presented above, we can state the following:

1. By using temporary storage, solutions with improved routing cost were obtained for the standard libraries of problems available in the literature for the VRP-SD. However, since not all VRP-TS and VRP-SDTS instances could be solved to optimality, further analysis is needed to establish strong conclusions. Thus, we propose to develop tailored solution approaches to solve both VRP-TS and VRP-SDTS to optimality.

2. By analysing solutions of the VRP-SDTS, it can seen that temporary storage is used in a large number of instances.

3. By modeling the VRP-TS using a two-index formulation, the number of nodes increases from n to $2(n-1)+1$, and thus, this problem becomes more difficult to solve when compared VRP-SDTS, where the number of nodes is n.

Chapter 8

Concluding remarks and directions for future research

8.1 Concluding remarks

In this dissertation, we have modeled, analyzed, and developed exact algorithms for some biomass logistics supply chain design and routing problems. Chapter 1 presented background material on biomass logistics and routing problems, and it introduced each of the problems addressed in this dissertation as well as their overreaching relationship.

In Chapter 2, we have introduced the Biomass Logistics Supply Chain Design Problem (BL-SCDP), a new and challenging problem that integrates the single-item capacitated lot-sizing problem with multiple parallel facilities (CLSP-MF), time-dependent selective high-multiplicity multiple travelling salesmen problem (TDS-HMmATSP), and biomass logistics (BL). A novel branch-and-price-based (B&P) method was developed to solve this problem that relies on determination of fast and effective upper bound, generation of lower bounds even if the subproblem is not solved to optimality, use of advanced start for subproblem and master problem, dual stabilization, and column management. Our implementation of this approach to the large-sized problem instances inspired by a real-life case study reveals its ability to obtain near-optimal solutions to problem instances

for which a direct application of CPLEX to their BL-SCDP model formulation fails to generate a feasible solution. The average optimality gap obtained for these problems by the proposed method is 4.99%. For those problem instances that CPLEX is able to generate a solution, the proposed B&P outperforms CPLEX. The average optimality gap obtained by the proposed method for these instances is 3.2% as against 10.7% for CPLEX.

In Chapter 3, we have addressed a dynamic corn stover harvest scheduling problem (CSHSP), where expensive custom harvesters must be assigned to harvest corn stover production fields that become available dynamically over the planning horizon. We have proposed a mixed-integer programming formulation to model the static CSHSP, and have proposed an advance-start solution in order to reduce the computational effort taken by CPLEX to solve this problem. We have illustrated the use of the proposed methodology on a dataset collected during the 2014 corn stover harvest season in Iowa, USA, and the results indicate that the static CSHSP was unable to be solved directly by CPLEX, while CPLEX with our advanced-start obtained solutions to these problems instances with an optimality gap less than 1%. Furthermore, we have performed a sensitivity analysis on field ready times, harvester capacities, and field due dates. From the results, we conclude that a change in field ready times impact the number of balers required for the corn stover harvest season. A decrement in the capacity of the custom harvesters, increases both the number of balers and the total cost incurred, while reducing the width of harvesting time windows increases the total significantlyy. Furthermore, we also have proposed a periodic reoptimization scheme to tackle the dynamic CSHSP, in which the static CSHSP is solved at different decision points throughout the harvest season. Results indicate that the proposed approach for the dynamic CSHSP generated solutions with a cost 21.0% higher than the cost obtained by the static approach, where field ready times are assumed to be known forehand.

In Chapter 4, we have used the idea introduced in Dantzig et al. (1954), Miliotis (1976), and Miliotis (1978) to generate subtour elimination constraints (SECs) only from integer points to solve large-sized single and multiple asymmetric traveling salesmen problems (ATSP and mATSP). For the ATSP, our proposed approach outperformed the direct solution by the commercial solver (CPLEX 12.5.1) of the three most effective polynomial-length formulations reported in Roberti and

Toth (2012) (MTZ, GG, and DL). In addition, our results indicated that the proposed algorithm is competitive with the most effective branch-and-cut algorithms reported in the literature. An additional advantage of our approach over the most effective existing exact approaches for the ATSP as described in Carpaneto et al. (1995), Fischetti and Toth (1992), Fischetti et al. (2003), and Applegate et al. (2006)) is that it can be easily implemented using an off-the-shelf optimization software package such as CPLEX. For the mATSP, the proposed approaches outperformed the direct solution by the commercial solver (CPLEX 12.5.1) of the three most effective polynomial-length formulations reported in Sarin et al. (2014a) (MTZ, GG, and BK), and they solved all mATSP instances of size greater than 443 cities with a maximum optimality gap of 2.18%.

In Chapter 5, we have studied the high-multiplicity asymmetric traveling salesman problem (HMATSP), which is an extension of the well-known traveling salesman problem in which cities are visited multiple times. This problem arises during the production of batches of different product types on one or more machines, where the machines (salesmen) process the product batches (cities) for a specified number of times while requiring sequence-dependent set-up costs. We have presented a single-commodity flow-based formulation for the HMATSP, and the results revealed that it is more effective despite having a weaker linear programming relaxation, where it attained optimal solutions to several instances for which the existing compact formulation was unable to generate a feasible solution. We have also developed Benders decomposition-based solution methods for both the proposed and the existing compact formulation as well as have used acceleration techniques to improve their convergence. The proposed accelerated Benders algorithms were able to solve HMATSP instances with up to 1001 cities within one hour of CPU time, which are the largest-sized instances solved in the literature. Furthermore, we have proposed several new variants or extensions of the HMATSP that arise in different production and routing applications. These pertain to the use of multiple salesmen located at a single base or at multiple bases. We likewise employed accelerated Benders decomposition-based exact solution approaches for these extensions, which were shown to solve instances with up to 1001 cities and to significantly outperform the direct solution of their model formulations by CPLEX (version 12.5.1).

In Chapter 6, we have addressed the fixed-destination multi-depot traveling salesman problem

with load balancing (FD-MTSPB), wherein m salesmen who are distributed across D depots must collectively visit a set of cities and return to their respective starting depots, and where the number of cities visited in each tour must lie within a pre-specified range. We have presented a new flow-based formulation for this problem, and have investigated its performance against two existing formulations. This formulation is neither the tightest nor the fastest when solved directly using CPLEX (12.5.1); however, it gives the best results when implemented in a Benders decomposition-based approach. We also developed a two-phase approach, which relies on solving a relatively simple relaxation of the problem (the non-fixed destination MTSPB) in the initial phase in order to derive an optimal solution to FD-MTSPB. In our computational experiments, the proposed two-phase approach outperformed the direct application of CPLEX to the three model formulations studied, and also outperformed the proposed Benders decomposition-based approaches for the model described in the literature as well as the ones developed in this chapter. Furthermore, we introduced an extension called the high-multiplicity FD-MTSPB (HMFD-MTSPB), where the cities are required to be visited multiple times, and we have presented two novel flow-based formulations for this problem that can be solved effectively using commercial off-the-shelf optimization packages. In addition, we have extended our proposed two-phase approach for solving HMFD-MTSPB, and have demonstrated this procedure to outperform the direct application of CPLEX for solving the alternative model formulations.

In Chapter 7, we have introduced the vehicle routing problem with temporary storage (VRP-TS), an extension of the traditional vehicle routing problem, where customer locations can be utilized as temporary storages. We also have studied the VRP-SDTS, a further extension of the VRP that comprises both the split delivery (SD) and temporary storage (TS) features. We have presented mixed-inter programming formulations for the VRP-TS and VRP-SDTS, and have investigated the benefits achieved by both of these features in reducing the routing cost over those for the traditional VRP and VRP with split delivery (VRP-SD). In particular, we have shown that the potential savings of VRP-TS over VRP is at least as much as obtained by the VRP-SD over VRP. Results revealed that by using temporary storage, solutions with lower cost can be obtained for the standard libraries of problems available in the literature for VRP-SD. Moreover, by analysing solutions obtained for VRP-SDTS, we observed that temporary storage was used several times.

8.2 Directions for future research

In Chapter 2, we have introduced the Biomass Logistics Supply Chain Design Problem (BL-SCDP) that focuses on determining where to locate and route load-out equipments to load biomass onto racks that will be hauled by tractor-trailers to a bio-energy plant over the planning horizon. In this model, we assume the number and location of SSLs to be known *a priori*. We suggest for future research to develop an integrated model that includes these decision as well into the BL-SCDP problem. Furthermore, we propose a polyhedral analysis of the subproblem of the branch-and-price method at theoretical level in order to develop facet defining inequalities and a customized solution approach.

In Chapter 3, we have addressed a corn stover harvesting scheduling problem (CSHSP), where parameter values are assumed to be known with certainty. However, because of uncertainty the estimation of these values parameters might be difficult to forecast. Therefore, we propose investigation of a stochastic model to include uncertainty mainly due to different weather scenarios.

In Chapters 4 and 5, we have used the idea of generating subtour elimination constraints (SECs) from integer points to solve the ATSP, mATSP, and HMATSP. For future research, we propose to investigate adaptations of the proposed approaches for solving different extensions of the ATSP, mATSP, HMATSP as well as their application to other routing problems. Besides, this approach might be useful to the researchers and practitioners in the context of applying ATSP, mATSP, HMATSP either as stand-alone models or as sub-models within some application settings.

In Chapter 7, we have introduced the vehicle routing problem with temporary storage (VRP-TS). At this point there are several open questions that need to be addressed in a future work. These are as follows: (i) is the worst-case bound between the VRP and VRP-TS greater than 2? (we have shown by an example that this bound is at least 2), and (ii) what is the worst-case bound between the VRP and VRP-SDTS?. We also suggest theoretical investigations of the VRP-TS and VRP-SDTS, and development of effective exact algorithms for their solution.

Bibliography

An, H., Wilhelm, W. E., and Searcy, S. W. (2011). A mathematical model to design a lignocellulosic biofuel supply chain system with a case study based on a region in central texas. *Bioresource Technology*, 102(17):7860–7870.

Applegate, D. L., Bixby, R. E., Chvatal, V., and Cook, W. J. (2006). *The traveling salesman problem: a computational study*. Princeton university press.

Archetti, C., Bianchessi, N., and Speranza, M. G. (2014). Branch-and-cut algorithms for the split delivery vehicle routing problem. *European Journal of Operational Research*, 238(3):685 – 698.

Archetti, C., Savelsbergh, M. W. P., and Speranza, M. G. (2006). Worst-case analysis for split delivery vehicle routing problems. *Transportation Science*, 40(2):226–234.

Ascheuer, N., Fischetti, M., and Grtschel, M. (2000). A polyhedral study of the asymmetric traveling salesman problem with time windows. *Networks*, 36(2):69–79.

Atallah, M. and Kosaraju, S. (1988). Efficient solutions to some transportation problems with applications to minimizing robot arm travel. *SIAM Journal on Computing*, 17(5):849–869.

Baker, T. and Muckstadt, J. (1989). The CHESproblems. *Technical Paper, Chesapeake Decision Sciences, Inc., Providence, RI (1989).*

Bektaş, T. (2006). The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219.

Bektaş, T. (2012). Formulations and Benders decomposition algorithms for multidepot salesmen problems with load balancing. *European Journal of Operational Research*, 216(1):83–93.

Belenguer, J. M., Martinez, M. C., and Mota, E. (2000). A lower bound for the split delivery vehicle routing problem. *Operations Research*, 48(5):801–810.

Bordenave, C., Gendreau, M., and Laporte, G. (2012). A branch-and-cut algorithm for the preemptive swapping problem. *Networks*, 59(4):387–399.

Burger, M. (2014). Exact and compact formulation of the fixed-destination travelling salesman problem by cycle imposement through node currents. In *Operations Research Proceedings 2013*, pages 83–88. Springer.

Calvo, R. W. (2000). A new heuristic for the traveling salesman problem with time windows. *Transportation Science*, 34(1):113–124.

Carpaneto, G., Dell'Amico, M., and Toth, P. (1995). Exact solution of large-scale, asymmetric traveling salesman problems. *ACM Trans. Math. Softw.*, 21(4):394–409.

Charikar, M., Khuller, S., and Raghavachari, B. (2001). Algorithms for capacitated vehicle routing. *SIAM Journal on Computing*, 31(3):665–682.

Chen, S., Golden, B., and Wasil, E. (2007). The split delivery vehicle routing problem: Applications, algorithms, test problems, and computational results. *Networks*, 49(4):318–329.

Cirasella, J., Johnson, D. S., McGeoch, L. A., and Zhang, W. (2001). *Algorithm Engineering and Experimentation*, chapter The Asymmetric Traveling Salesman Problem: Algorithms, Instance Generators, and Tests, pages 32–59.

Cortés, C. E., Matamala, M., and Contardo, C. (2010). The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. *European Journal of Operational Research*, 200(3):711–724.

Cosmadakis, S. S. and Papadimitriou, C. H. (1984). The traveling salesman problem with many visits to few cities. *SIAM Journal on Computing*, 13(1):99–108.

Cundiff, J., Grisso, R., and Ravula, P. (2004). Management system for biomass delivery at a conversion plant. In *In: ASAE. ASAE/CSAE, St. Joseph, MI*, page 46169.

Cundiff, J. S., Dias, N., and Sherali, H. D. (1997). A linear programming approach for designing a herbaceous biomass delivery system. *Bioresource Technology*, 59(1):47–55.

D'Ambrosio, C., Lodi, A., and Martello, S. (2010). *Combinatorial Traveling Salesman Problem Algorithms*. John Wiley & Sons, Inc.

Dantzig, G., Fulkerson, R., and Johnson, S. (1954). Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, 2(4):393–410.

Desaulniers, G., Desrosiers, J., and Solomon, M. (2005). *Column generation*. Springer, New York.

Desrochers, M. and Laporte, G. (1991). Improvements and extensions to the miller-tucker-zemlin subtour elimination constraints. *Oper. Res. Lett.*, 10(1):27–36.

Dror, M. and Trudeau, P. (1990). Split delivery routing. *Naval research logistics*, 37(3):383–402.

du Merle, O., Villeneuve, D., Desrosiers, J., and Hansen, P. (1999). Stabilized column generation. *Discrete Mathematics*, 194(13):229–237.

Dumas, Y., Desrosiers, J., Gelinas, E., and Solomon, M. M. (1995). An optimal algorithm for the traveling salesman problem with time windows. *Operations Research*, 43(2):367–371.

DuPont (2015). The dupont cellulosic ethanol facility in nevada, iowa: Leading the way for commercialization. http://www.dupont.com/products-and-services/industrial-biotechnology/advanced-biofuels/cellulosic-ethanol/nevada-iowa-cellulosic-ethanol-plant.html.

Eksioglu, S. D., Acharya, A., Leightley, L. E., and Arora, S. (2009). Analyzing the design and management of biomass-to-biorefinery supply chain. *Computers & Industrial Engineering*, 57(4):1342–1352.

Erdoğan, G., Battarra, M., and Calvo, R. W. (2015). An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. *European Journal of Operational Research*, 245(3):667 – 679.

Fales, S., Hess, J., Wilhelm, W., Erbach, D., Provine, W., Vogel, K., Peterson, T., and Runge, E. (2008). Convergence of agriculture and energy: Ii. producing cellulosic biomass for biofuels. *Engineering and Technology for Sustainable World*, 15(2):10–11.

Feillet, D., Dejax, P., and Gendreau, M. (2005). Traveling salesman problems with profits. *European Journal of Operational Research*, 39(2):188–205.

Fischetti, M., Lodi, A., and Toth, P. (2002). Exact methods for the asymmetric traveling salesman problem. In Gutin, G. and Punnen, A. P., editors, *The Traveling Salesman Problem and Its Variations*, chapter 4, pages 169–205. Springer US, Boston, MA.

Fischetti, M., Lodi, A., and Toth, P. (2003). *Combinatorial Optimization — Eureka, You Shrink!: Papers Dedicated to Jack Edmonds 5th International Workshop Aussois, France, March 5–9, 2001 Revised Papers*, chapter Solving Real-World ATSP Instances by Branch-and-Cut, pages 64–77. Springer Berlin Heidelberg, Berlin, Heidelberg.

Fischetti, M., Salvagnin, D., and Zanette, A. (2010). A note on the selection of benders' cuts. *Mathematical Programming*, 124(1):175–182.

Fischetti, M. and Toth, P. (1992). An additive bounding procedure for the asymmetric travelling salesman problem. *Mathematical Programming*, 53(1):173–197.

Fox, K. R., Gavish, B., and Graves, S. C. (1980). An n-constraint formulation of the (time-dependent) traveling salesman problem. *Operations Research*, 28(4):1018–1021.

Gavish, B. and Graves, S. C. (1978). The travelling salesman problem and related problems. Working paper GR-078-78. Cambridge, MA: Operation Research Center, Massachusetts Institute of Technology.

Gendreau, M., Laporte, G., and Semet, F. (1998). A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32(4):263–273.

Golden, B. L., Magnanti, T. L., and Nguyen, H. Q. (1977). Implementing vehicle routing algorithms. *Networks*, 7(2):113–148.

Gouveia, L. and Voß, S. (1995). A classification of formulations for the (time-dependent) traveling salesman problem. *European Journal of Operational Research*, 83(1):69–82.

Gribkovskaia, I. and Laporte, G. (2008). One-to-many-to-one single vehicle pickup and delivery problems. In Golden, B., Raghavan, S., and Wasil, E., editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer US.

Grigoriev, A. and van de Klundert, J. (2006). On the high multiplicity traveling salesman problem. *Discrete Optimization*, 3(1):50 – 62.

Grisso, R. D., McCullough, D., Cundiff, J. S., and Judd, J. D. (2013). Harvest schedule to fill storage for year-round delivery of grasses to biorefinery. *Biomass and Bioenergy*, 55(0):331–338.

Guan, D. J. and Zhu, X. (1998). Multiple capacity vehicle routing on paths. *SIAM Journal on Discrete Mathematics*, 11(4):590.

Huang, Y., Chen, C.-W., and Fan, Y. (2010). Multistage optimization of the supply chains of biofuels. *Transportation Research Part E: Logistics and Transportation Review*, 46(6):820–830.

IBM (2016). Benders decomposition using lazy constraint and user cut callbacks.

Judd, J. D., Sarin, S. C., and Cundiff, J. S. (2012). Design, modeling, and analysis of a feedstock logistics system. *Bioresource Technology*, 103(1):209–218.

Kara, I. and Bektaş, T. (2006). Integer linear programming formulations of multiple salesman problems and its variations. *European Journal of Operational Research*, 174(3):1449 – 1458.

Kerivin, H., Lacroix, M., and Mahjoub, A. (2012). Models for the single-vehicle preemptive pickup and delivery problem. *Journal of Combinatorial Optimization*, 23(2).

Kerivin, H., Lacroix, M., Mahjoub, A., and Quilliot, A. (2008). The splittable pickup and delivery problem with reloads. *European Journal of Industrial Engineering*, 2(2):112–133.

Laporte, G. (1992). The traveling salesman problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(2):231 – 247.

Laporte, G. and Martello, S. (1990). The selective travelling salesman problem. *Discrete Applied Mathematics*, 26(2-3):193–207.

Laporte, G., Nobert, Y., and Taillefer, S. (1988). Solving a family of multi-depot vehicle routing and location-routing problems. *Transportation Science*, 22(3):161–172.

Lin, T., Rodriguez, L. F., Shastri, Y. N., Hansen, A. C., and Ting, K. (2014). Integrated strategic and tactical biomass biofuel supply chain optimization. *Bioresource Technology*, 156(0):256–266.

Liu, J., Grisso, R., and Cundiff, J. (2013). *Harvest Systems and Analysis for Herbaceous Biomass*. InTech.

Magnanti, T. L. and Wong, R. T. (1981). Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484.

Masson, R., Lehuédé, F., and Péton, O. (2013). An adaptive large neighborhood search for the pickup and delivery problem with transfers. *Transportation Science*, 47(3):344–355.

Masson, R., Lehuédé, F., and Péton, O. (2014). The dial-a-ride problem with transfers. *Computers & Operations Research*, 41(0):12–23.

Miliotis, P. (1976). Integer programming approaches to the travelling salesman problem. *Mathematical Programming*, 10(1):367–378.

Miliotis, P. (1978). Using cutting planes to solve the symmetric travelling salesman problem. *Mathematical Programming*, 15(1):177–188.

Miller, C. E., Tucker, A. W., and Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *J. ACM*, 7(4):326–329.

Mitrović-Minić, S., Krishnamurti, R., and Laporte, G. (2004). Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B: Methodological*, 38(8):669 – 685.

Mitrovic-Minic, S. and Laporte, G. (2006). The pick up and delivery problem with time windows and transshipment. *Information Systems and Operational Research*, 44(3):216–227.

Öncan, T., Altnel, İ. K., and Laporte, G. (2009). A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, 36(3):637 – 654.

Padberg, M. and Rinaldi, G. (1990). An efficient algorithm for the minimum capacity cut problem. *Mathematical Programming*, 47(1):19–36.

Papadakos, N. (2008). Practical enhancements to the Magnanti-Wong method. *Operations Research Letters*, 36(4):444–449.

Perlack, R. D., Wright, L. L., Turhollow, A. F., Graham, R. L., Stokes, B. J., and Erbach, D. C. (2005). Biomass as feedstock for a bioenergy and bioproducts industry: the technical feasibility of a billion-ton annual supply.

Pesant, G., Gendreau, M., Potvin, J.-Y., and Rousseau, J.-M. (1998). An exact constraint logic programming algorithm for the traveling salesman problem with time windows. *Transportation Science*, 32(1):12–29.

Pferschy, U. and Staněk, R. (2016). Generating subtour elimination constraints for the tsp from pure integer solutions. *Central European Journal of Operations Research*, pages 1–30.

Picard, J.-C. and Queyranne, M. (1978). The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, 26(1):86–110.

Pinedo, M. L. (2012). *Scheduling: theory, algorithms, and systems*. Springer Science & Business Media.

POET-DSM (2014). First commercial-scale cellulosic ethanol plant in the U.S. opens for business. http://poet-dsm.com/pr/first-commercial-scale-cellulosic-plant.

Qu, Y. and Bard, J. F. (2012). A {GRASP} with adaptive large neighborhood search for pickup and delivery problems with transshipment. *Computers & Operations Research*, 39(10):2439 – 2456.

Rais, A., Alvelos, F., and Carvalho, M. (2014). New mixed integer-programming model for the pickup-and-delivery problem with transshipment. *European Journal of Operational Research*, 235.

Reinelt, G. (1994;2003;). *The traveling salesman: computational solutions for TSP applications*, volume 840.;840;. Springer-Verlag, New York;Berlin;.

Resop, J. P., Cundiff, J. S., and Heatwole, C. D. (2011). Spatial analysis to site satellite storage locations for herbaceous biomass in the piedmont of the southeast. *Applied Engineering in Agriculture*, 27(1):25–32.

Roberti, R. and Toth, P. (2012). Models and algorithms for the asymmetric traveling salesman problem: an experimental comparison. *EURO Journal on Transportation and Logistics*, 1(1):113–133.

Salazar-González, J.-J. and Santos-Hernández, B. (2015). The split-demand one-commodity pickup-and-delivery travelling salesman problem. *Transportation Research Part B: Methodological*, 75:58 – 73.

Sambasivan, M. and Yahya, S. (2005). A lagrangean-based heuristic for multi-plant, multi-item, multi-period capacitated lot-sizing problems with inter-plant transfers. *Computers & Operations Research*, 32(3):537–555.

Sarin, S. C., Sherali, H. D., Judd, J. D., and Tsai, P.-F. J. (2014a). Multiple asymmetric traveling salesmen problem with and without precedence constraints: Performance comparison of alternative formulations. *Computers & Operations Research*, 51(0):64–89.

Sarin, S. C., Sherali, H. D., and Liao, L. (2014b). Primary pharmaceutical manufacturing scheduling problem. *IIE Transactions*, 46(12):1298–1314.

Sarin, S. C., Sherali, H. D., and Yao, L. (2011). New formulation for the high multiplicity asymmetric traveling salesman problem with application to the chesapeake problem. *Optimization Letters*, 5(2):259–272.

Shang, J. S. and Cuff, C. K. (1996). Multicriteria pickup and delivery problem with transfer opportunity. *Computers & Industrial Engineering*.

Sherali, H. D. and Lunday, B. J. (2013). On generating maximal nondominated Benders cuts. *Annals of Operations Research*, 210(1):57–72.

Sokhansanj, S. and Turhollow, A. (2002). Baseline cost for corn stover collection. *Applied Engineering in Agriculture*, 18(5):525.

Sung, C. S. (1986). A single-product parallel-facilities production-planning model. *International Journal of Systems Science*, 17(7):983–989.

Tembo, G., Epplin, F. M., and Huhnke, R. L. (2003). Integrative investment appraisal of a lignocellulosic biomass-to-ethanol industry. *Journal of Agricultural and Resource Economics*, 28(3):611–633.

TSPLIB (1997). A library of traveling salesmen and related problem instances.

Vansteenwegen, P., Souffriau, W., and Oudheusden, D. V. (2011). The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10.

Xie, F., Huang, Y., and Eksioglu, S. (2014). Integrating multimodal transport into cellulosic biofuel supply chain design under feedstock seasonality with a case study based on california. *Bioresource Technology*, 152(0):15–23.

Zangwill, W. I. (1966). A deterministic multi-period production scheduling model with backlogging. *Management Science*, 13(1):105–119.

Zhang, J., Osmani, A., Awudu, I., and Gonela, V. (2013). An integrated optimization model for switchgrass-based bioethanol supply chain. *Applied Energy*, 102(0):1205–1217.

Zhu, X., Li, X., Yao, Q., and Chen, Y. (2011). Challenges and models in supporting logistics system design for dedicated-biomass-based bioenergy industry. *Bioresource Technology*, 102(2):1344–1351.

Zhu, X. and Yao, Q. (2011). Logistics system design for biomass-to-bioenergy industry with multiple types of feedstocks. *Bioresource Technology*, 102(23):10936–10945.