The Pennsylvania State University

The Graduate School

College of Engineering

FLEET PLANNING

IN THE CAR RENTAL BUSINESS

A Dissertation in

Industrial Engineering and Operations Research

by

Gen-Han Wu

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2010

The dissertation of Gen-Han Wu was reviewed and approved* by the following:

Tom M. Cavalier
Professor of Industrial Engineering
Dissertation Advisor
Chair of Committee

A. Ravi Ravindran
Professor of Industrial Engineering

David A. Nembhard
Associate Professor of Industrial Engineering

Terry P. Harrison
Professor of Supply Chain and Information System

Paul Griffin
Professor of Industrial Engineering
Head of the Harold and Inge Marcus Department
of Industrial and Manufacturing Engineering

*Signatures are on file in the Graduate School.

Abstract

In the United States, consumption capacity in tourism is poised to grow significantly, and transportation is one of the essential elements in tourism. Airplanes and car rentals are the most typical modes of public travel. While air transport has earned significant scholarly attention and there exist abundant studies, fleet planning in the car rental business is discussed only minimally in operations research. Therefore, the purpose of this research is not only to build a thorough analytical framework for car rental fleet planning in different time phases, but also to develop practical algorithmic procedures.

In long-term planning, pool segmentation and hub selection are studied. All rental locations are split into different pools and a hub is selected within each pool. The proposed clustering-based iterative algorithm offers a reliable clustering method to quickly find an initial solution with a small solution gap and an iterative method to gradually approach a near-optimum. In mid-term planning, inter-pool moves and asset replacement are distributed among different pools based on the change of seasonal demand. Numerical results have shown that the best-improvement descent local search with the structure of better neighbors has very good performance and can obtain a satisfactory solution in an extremely short time. In short-term planning, vehicle imbalance at different locations forces empty vehicles to be redistributed. Daily planning of demand allocation and empty flow redistribution is addressed in the same pool. Car upgrade policy and service level are also considered. A first-improvement descent local search is developed. Computing results demonstrate that the first-improvement descent local search not only obtains relatively good solutions in quite a short time but also solves very large scale integer programming problems easily.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor, Dr. Tom M. Cavalier, for his continued guidance over the past years. Without his patience and constructive advice, I could not have completed this dissertation. I would also like to thank my committee members, Dr. A. Ravi Ravindran, Dr. David A. Nembhard, and Dr. Terry P. Harrison, for their valuable comments and suggestions that make my dissertation more complete.

Finally, I would like to express my profound gratitude to my parents. None of this would have been possible without their love, patience, support, and encouragement.

CHAPTER 1

INTRODUCTION

1.1 Research Origin and Motivation

In the United States, consumption capacity in tourism is poised to grow significantly. Within tourism, transportation is one of the essential elements and the most typical modes of public travel are airplanes and car rentals. International and long-distance travel relies on airplanes, but post-flight excursions and trips depend upon car rentals.

In order to rent a car, consumers pay a car rental company a fee to rent an automobile for a short period of time, ranging from a few hours to several weeks. A car rental company is comprised of numerous local branches located within airports or throughout urban areas. Not only do car rental companies offer online reservations, but also many on-line travel agencies including Priceline, Expedia, and others, offer price comparisons between companies. Car rental companies primarily serve customers who make excursions or business trips and they offer customers a choice of economy, compact, medium, or luxury cars to meet customers' needs. Therefore, car rental companies presumably own a large number of fleet vehicles, and their rental facilities are widely distributed to easily schedule the vehicles. Moreover, car rental companies enter into agreements with vehicle manufacturers, including Ford, Chrysler, and Dodge, to purchase cars at extremely low prices. As such, larger car rental companies are generally more competitive than smaller ones

because these larger companies have more rental facilities, more fleet vehicles, and more car types, and therefore can decrease operational and maintenance costs and provide their customers a convenient, cheap, and diverse car rental environment.

While air transport has earned significant scholarly attention (Doy and Pope 1985; Jenkins 1987; Marker 1991; Aykin 1995; Goodovitch 1996; Orlady 2002; Barnhart *et al.* 2003; Janic 2003;Donohue 2006; Khoury *et al.* 2007; Baxley *et al.* 2008) and there exist abundant studies on the planning of flight routes (Yan and Young 1996; Rosenthal and Walsh 1996; Barnett 2000; Hsu and Wen 2000), flight schedules (Jarrah *et al.* 1993; Yan and Tu 1997; Cao and Kanafani 2000), flight fares(Chatwin 1996; Chatwin 1998; Nero and Black 1998; Subramanian *et al.* 1999), and aviation rights(Stickle *et al.* 1991; Dodgson 1994), fleet planning in the car rental business(Pachon 2000) is discussed only minimally in operations research even though car rental fleet planning possesses economic value. Moreover, in the past, most of the effort in car rental fleet planning was devoted to small scale problems. Hence, more effort and research in car rental fleet planning is essential to continued improvement in the competitive environment.

1.2 Research Purpose

Since more effort in car rental fleet planning is essential, the purpose of this study is not only to build a thorough analytical framework for car rental fleet planning in different time phases, but also to develop practical algorithmic procedures, which can solve the problems in actual problem sizes. Ultimately, these practical algorithms will build a complete, rapid, and accurate plan of vehicle distribution in order to effectively reduce staff misjudgment and workloads in the car rental company.

1.3 Problem Statement

In order to successfully manage the fleet planning of a large car rental company, the first step is to assess the market, so that the demand can be carefully evaluated and forecasted. The yearly demand and profit potential in different parts of the country help determine the locations of rental facilities as well as the number of initial fleet vehicles needed. Due to the large number of rental facilities, the locations are split into several pools and one location from each pool is designated as the hub of the pool. The hub is responsible for seasonal vehicle distribution, depreciation, and procurement among different pools. To manage rental facilities within the same region, car allocation and empty vehicle redistribution need to be considered. Moreover, when the reserved car type is not available, it needs to be upgraded without an additional cost to the customer. Since a car rental company charges for the reserved car type's expense even if upgrading to another car type, the upgrade policy is an important task in the problem of car rental operation.

Focusing on the above issues, this study develops three fleet operation plans: long-term, mid-term, and short-term. In long-term planning, this study assumes that the location and the yearly demand of each rental facility are known. The rental facilities are split into different pools based on yearly demand, distance, hub costs, and car upgrade policy. Additionally, a clustering-based iterative algorithm is devised to find a suitable solution. In mid-term planning, a hub represents each pool, and seasonal demand is distributed among different pools, which are called inter-pool moves. This study also considers the procurement of new cars, the depreciation of old cars, and car upgrade policy, and develops a best-improvement descent local search, embedded with the structure of better neighbors, to obtain satisfactory

solutions. In short-term planning, daily demand is assumed to be known and a car upgrade policy is employed. In the same pool, unused cars can be redistributed on the same night. A first-improvement descent local search is proved to be very effective in solving this problem. The overall problem related to car rental operations is outlined in Figure 1.1.

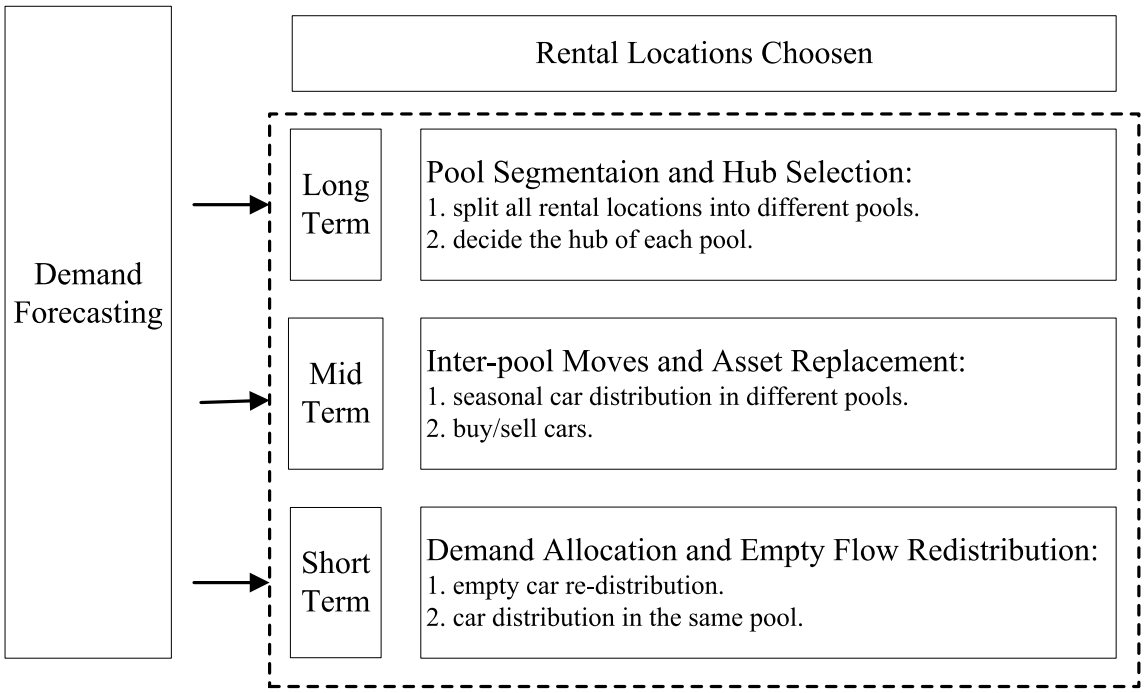

Figure 1.1. The problem of car rental operations

1.4 Dissertation Framework

This dissertation is organized into seven chapters. Chapter 2 provides the car rental business profile in the United States by discussing the history of the development of the car rental business, the statistics of the market, and the functions of car rental software. Chapter 3 discusses car rental problems and fleet planning literature. Then, Chapter 4 introduces the model of pool segmentation and hub selection. A clustering-based iterative algorithm is proposed and is proven to be very effective. Chapter 5 provides an overview of mid-term fleet planning, which deals with inter-pool moves and asset replacement. A best-improvement descent local search is developed and validated. The short-term fleet planning, which deals with demand allocation and empty flow redistribution, is introduced in Chapter 6. An analytical model is developed and a first-improvement descent local search is used to solve this problem. Finally, Chapter 7 outlines the concluding remarks and presents directions for further extensions and research. An overview of the research framework of this dissertation is shown in Figure 1.2.

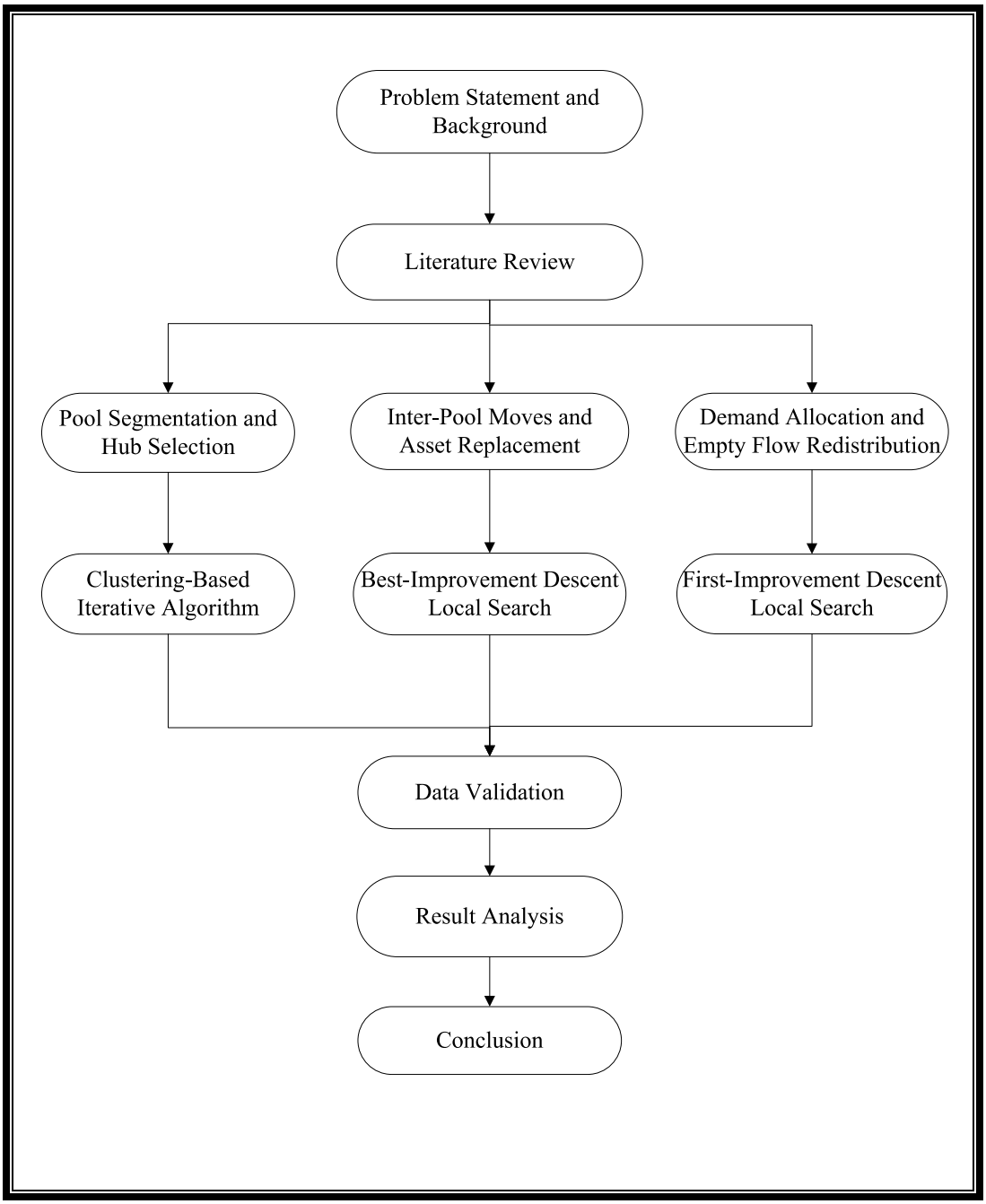

Figure 1.2. Overview of the research framework

CHAPTER 2

CAR RENTAL BUSINESS PROFILE

2.1 History of the U.S. Car Rental Business

Although there is not much recorded detail about the origin of the first car rental company, many believe that the first car rental company was founded in 1916 by a Nebraskan native, Joe Saunders, who rented his Ford Model T to a traveling businessman and charged 10 cents per mile of use. From these humble beginnings Saunders eventually achieved great car rental success. By 1925, Saunders owned rental businesses in 21 states. Unfortunately, due to the Great Depression, Saunders' business declared bankruptcy during the early 1930s.

However, Saunders was not the only businessman who seized the car rental opportunity. Another businessman, Walter L. Jacobs, started renting Ford Model T cars to his customers in 1918. By 1923 Jacobs experienced success of over $ 1 million in annual sales and attracted the interest of John Hertz, who was the owner of The Yellow Cab and Yellow Truck and Coach Manufacturing Company. Hertz acquired Jacobs' business and eventually sold his Hertz's Yellow Truck Company to General Motors in 1929. The car rental business became known as the Hertz Drive-Ur-Self System.

The reputation established previously in the car rental business was damaged during the Prohibition period because many believed that rental cars were frequently used by

bootleggers and robbers. After the 18th Amendment was repealed in 1933, the industry rebuilt a respectable reputation and grew considerably.

Later, the expansion of the rail networks bolstered the car rental business for a number of years. Many railway companies encouraged rental car use by allocating spaces for rental booths at railroad stations. In addition, rail companies allowed passengers to reserve cars in advance of their arrivals at stations using the telegraph network.

The car rental business grew rapidly after WWII because the boom of business travels in the airline industry meant that more people needed a car for their post-flight business trips. Car rental companies opened franchises at airports to allow passengers vehicular transport at their arrival destination. For example, Hertz developed the “fly-drive” car rental concept by opening franchises at Chicago’s Midway Airport in 1932. Avis centered all of its operations from airports and advertised services through the airline companies. The car rental industry has been very competitive since the 1960s because many small companies have competed for profits through small kiosks in tightly compressed airport space.

Another main car rental company, Enterprise Rent-A-Car, which was founded in 1962, adopted completely different advertising strategies by appealing to customers who needed a replacement car or did not have a car. Enterprise opened their facilities extensively in local spots and offered cheaper prices, but older cars to their customers. Its business model continues to earn great success.

When car rental companies began the practice of selling their old cars to the public in the 1970s and the 1980s, some large car rental companies, such as Hertz, became major used

car dealers as well. At the same time, to counter this competition, major vehicle manufacturing firms tried to buy used car companies in order to ensure these companies would primarily purchase their new cars. In fact, car rental companies accounted for about 10 percent of all domestic auto sales in the early 1990s.

The growth of the car rental business slowed down after the terrorist attacks on September 11, 2001. However, it has made a dramatic comeback in recent years. This renewed interest in car rentals results, at least in part, from the prevalence of the Internet, which has made online reservations easier than ever. In 2003, online bookings had become important for the industry, with 21% of the bookings coming from online agencies and 15% directly from the car rental companies' web sites. The importance of online bookings is expected to grow even further.

For companies to survive in this highly competitive industry, car rental companies must provide unrivalled service. For instance, car rental companies are increasingly employing satellite navigation systems for their customers. With prices increasingly forced down by rivals, the customer now faces possibly the widest choices of cars and affordable prices ever in the history of the car rental business.

2.2 The U.S. Car Rental Market

According to the statistics provided by Auto Rental News*, in the last 20 years, the size of the fleets and the market revenue in the U.S. car rental market has progressively increased (see Figure 2.1). Although this trend has leveled off and stabilized since the year 2000, overall the size of the car fleet grew from 790,000 in 1986 to 1,813,000 in 2008. As a result of this growth, the market revenue has expanded from 9.7 billion dollars in 1991 to 21.88 billion dollars in 2008.

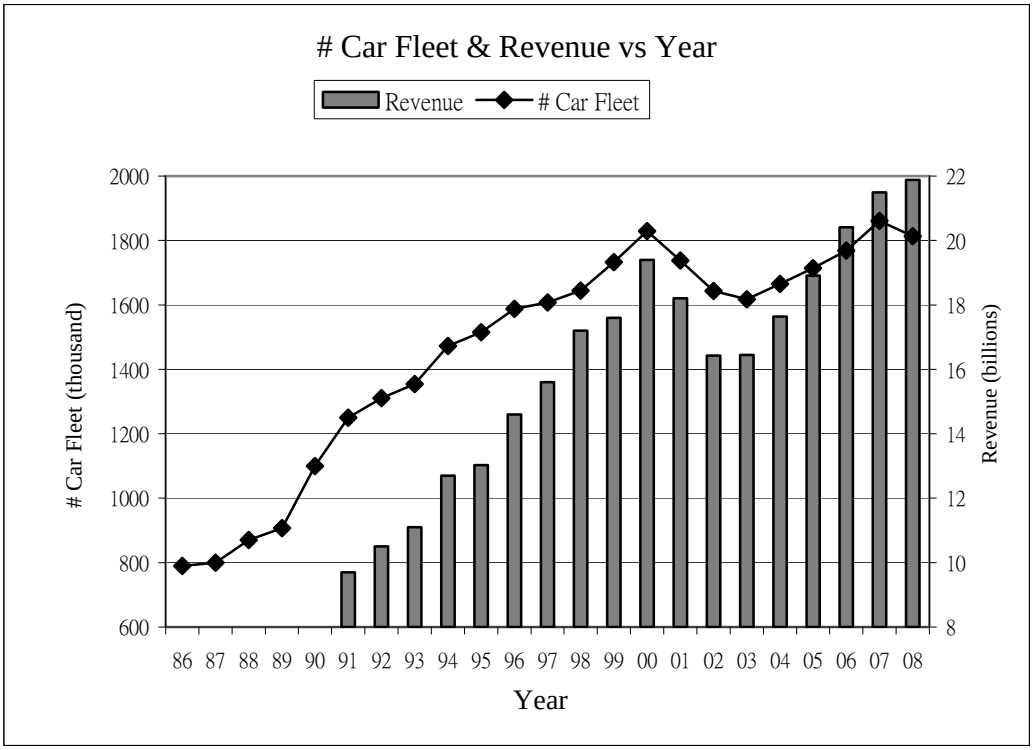

Figure 2.1. The U.S. car rental fleet size and market revenue

Note : * Auto Rental New http://www.autorentalnews.com/t_inside.cfm?action=statistics

Although the U.S. car rental market has been steady in recent years, there has been a rise in mergers and acquisitions. Enterprise, which is comprised of Enterprise Rent-A-Car, Alamo Rent A Car, and National Car Rental, has been the largest car rental company in North America since the late 1990's. Republic Service Inc. acquired Alamo Rent A Car in 1995 and in 1996 bought National Car Rental. Republic Service Inc., renamed AutoNation Inc. in 1998, spun off its unprofitable car rental unit ANC Rental Corp. in 2000. Vanguard Car Rental took over the bankrupt ANC Rental Corp. in 2003, and in 2007, Enterprise Rent-A-Car bought Vanguard Car Rental to consolidate its holdings in the U.S. auto rental business.

Avis Budget Group, which includes Avis Rent A Car and Budget Rent A Car, is the third largest car rental company after Enterprise Rent-A-Car and Hertz. HFS Incorporated acquired Avis Rent A Car in 1997, but Cendant Corp. merged with HFS Incorporated later that same year. In 2002, Cendant Corp. bought Budget Rent A Car and then approved a plan to separate Cendant into four independent companies, of which the car rental business became Avis Budget Group. Avis and Budget operate a shared fleet of cars and have the same back-end system. However, they operate at different locations, offer different service levels, and have different price structures.

Dollar Thrifty Automotive Group, which consists of Dollar Rent A Car and Thrifty Car Rental, is the fourth largest car rental company in North America. Thrifty Car Rental was acquired by Chrysler Corp. in 1989 and then Chrysler Corp. bought Dollar Rent A Car in 1990. In 1997, Chrysler announced that its rental car subsidiary, Dollar Thrifty Automotive

Group, was incorporated. The profiles of these car rental companies are obtained from the websites of the U.S. car rental companies and displayed in Table 2.1.

Table 2.1. Summary of the U.S. car rental companies

Company	Year Founded	Birthplace	Headquarters	Parent Company (Co-op Manufacturer)	
National	1947	Tulsa, OK	Tulsa, OK	Vanguard	Enterprise (GM)
Alamo	1974	Tampa, FL	Tulsa, OK		
Enterprise	1957	St. Louis, MO	St. Louis, MO	Enterprise	
Hertz	1918	Chicago, IL	Park Ridge, NJ	Hertz Group (Ford, GM)	
Avis	1946	Detroit, MI	Parsippany, NJ	Avis Budget Group (GM, Ford)	
Budget	1958	Los Angeles, CA	Chicago, IL		
Dollar	1965	Los Angeles, CA	Tulsa, OK	Dollar-Thrifty Automotive (Chrysler)	
Thrifty	1950	Tulsa, OK	Tulsa, OK		

Table 2.2 and Figure 2.2 show the number of cars and the percentage of the car fleet market share owned by the individual rental companies. Of all large car rental companies, only Enterprise Rent-A-Car grew rapidly in its size of car fleet, which increased from 460,100 in 2000 to 627,300 in 2008; its share in the car fleet market jumped from 25.1% to 34.6 % as well. Other car rental companies, however, continually shrank or remained stable over the last 7 years. In Figure 2.2, it is observed that the car rental companies can be separated into three groups based on the scale of the market. The first group merely includes Enterprise Rent-A-Car, which includes more than 30% of the market fleet. The second group includes Hertz, National/Alamo, Avis, Budget, and Dollar Thrifty. These companies utilize between 5% and 20% of the market fleet. The third group includes other medium-type companies, such as Advantage, U-Save, Payless, and others. Their market fleet is between

0.2% and 1%. In Table 2.2, other small rental companies shank from 90,000 in 2000 to 63,000 in 2008. These companies comprise no more than 5% of the market fleet in total.

Table 2.2. Summary of the number of cars by rental companies

Company	U.S. CARS IN SERVICE (Thousand)								
	2000	2001	2002	2003	2004	2005	2006	2007	2008
Enterprise Rent-A-Car	460.1	486.1	488.7	510.4	540.2	592.4	630.1	643.3	627.3
Hertz	350.0	320.0	304.0	315.0	315.0	315.0	290.0	327.2	311.0
National / Alamo	322.0	271.0	250.0	220.0	209.4	209.4	208.4	232.9	226.7
Avis Rent A Car	220.0	220.0	190.0	184.0	200.0	200.0	190.8	204.2	220.0
Budget Rent A Car	148.0	148.0	124.0	105.0	105.0	105.0	134.2	143.6	155.0
Dollar Thrifty Automotive	128.8	128.4	122.8	130.0	138.9	140.0	85.0	167.0	140.2
Advantage Rent-A-Car	13.0	15.0	15.0	15.5	15.5	15.0	17.0	20.0	15.0
U-Save Auto Rental	14.5	13.4	12.9	10.0	8.7	14.0	11.5	11.8	11.5
Payless Car Rental	7.5	8.5	8.8	9.2	8.5	10.0	10.0	10.0	10.0
ACE Rent A Car	8.0	8.0	7.2	10.0	12.0	12.5	11.5	9.0	9.0
Rent-A-Wreck	12.1	11.6	14.0	11.4	9.8	8.1	6.7	7.3	5.8
Triangle Rent-A-Car	4.2	3.9	4.4	4.0	4.6	4.8	6.0	6.0	5.5
Fox Rent A Car	N/A	N/A	N/A	N/A	5.2	6.2	6.8	8.7	8.7
Affordable/Sensible	N/A	N/A	N/A	N/A	4.8	5.2	5.0	5.0	4.0
Independent (3000+)	90.0	107.2	87.5	84.3	80.0	74.3	70.5	65.5	63.0
Totals	1,829.7	1,738.3	1,643.3	1,617.3	1,772.9	1,714.0	1,683.4	1,861.5	1,812.7

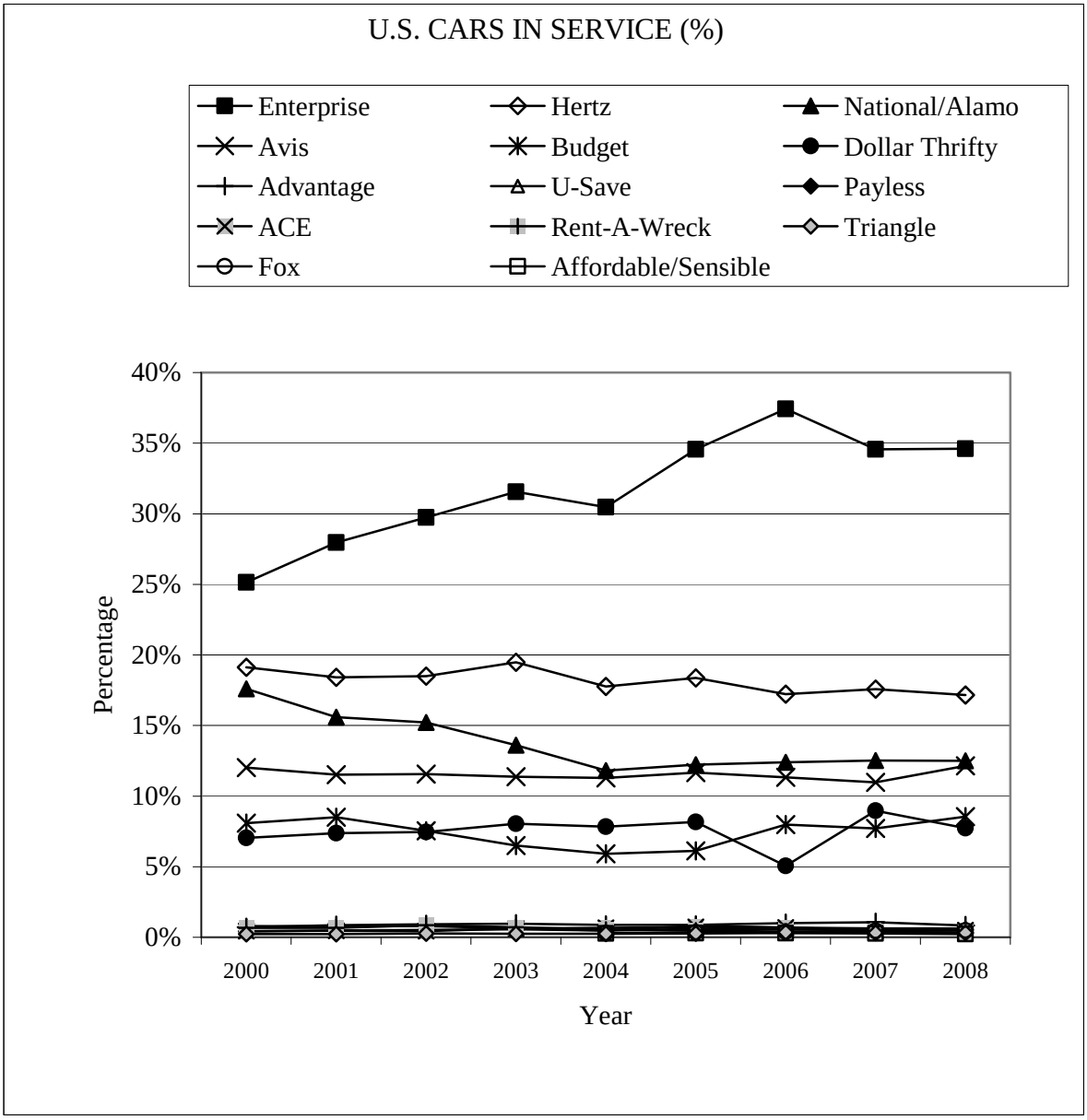

Figure 2.2. The percentage of car fleet market share by rental companies

The number of car rental facilities in different rental companies is shown in Table 2.3. Table 2.4 provides the car rental revenue for each rental company, and the percentage of car revenue market share of each rental company is given in Figure 2.3. Over the last 9 years, only Enterprise Rent-A-Car has grown rapidly in terms of the number of locations, rental revenue, and revenue market share.

Table 2.3. Summary of the number of the car rental facilities by rental companies

Company	# U.S. Locations								
	2000	2001	2002	2003	2004	2005	2006	2007	2008
Enterprise Rent-A-Car	4,018	4,398	4,708	4,987	5,388	5,719	6,019	6,131	6,159
Hertz	1,300	1,300	N/A	N/A	N/A	N/A	2,875	2,850	2920
National / Alamo	954	978	500	470	626	626	623	662	647
Avis Rent A Car	1,000	931	975	985	1,035	1,111	1,199	1,200	1,285
Budget Rent A Car	1,110	1,042	1,000	933	858	773	842	850	1,003
Dollar Thrifty Automotive	835	808	673	708	620	686	575	606	609
Advantage Rent-A-Car	150	150	150	150	150	132	100	108	45
U-Save Auto Rental	500	463	419	344	255	359	375	390	380
Payless Car Rental	78	85	85	91	78	43	46	41	41
ACE Rent A Car	39	39	44	56	82	90	85	85	101
Rent-A-Wreck	488	669	422	380	327	271	298	280	222
Triangle Rent A Car	20	20	21	21	25	25	28	30	30
Fox Rent A Car	N/A	N/A	N/A	N/A	20	39	28	29	29
Affordable/Sensible	N/A	N/A	N/A	N/A	240	246	250	225	210
Independent (3000+)	7,000	7,820	N/A	N/A	7,500	7,200	6,800	3,275	6,200
Totals	19,012	19,066	N/A	N/A	17,663	17,401	20,143	16,762	19,881

Table 2.4. Summary of the car rental revenue by rental companies

Company	U.S. Rental Revenue(million USD)								
	2000	2001	2002	2003	2004	2005	2006	2007	2008
Enterprise Rent-A-Car	$4500	$5100	$5250	$5490	$5830	$6400	$6800	$7100	$7,500
Hertz	$3980	$2900	$3050	$3110	$3500	$3650	$3770	$3,900	$3,860
National / Alamo	$3400	$3000	$2000	$1800	$1840	$1930	$2700	$2900	$2,900
Avis Rent A Car	$2400	$2380	$2250	$2140	$2280	$2700	$2900	$3100	$3,200
Budget Rent A Car	$1800	$1700	$1000	$940	$1130	$1400	$1500	$1,500	$1,600
Dollar Thrifty Automotive	$1480	$1430	$1500	$1560	$1680	$1390	$1460	$1,680	$1,650
Advantage Rent-A-CarI	$120	$120	$135	$139	$139	$150	$155	$220	$136
U-Save Auto Rental	$135	$148	$131	$118	$90	$95	$98	$102	$98
Payless Car Rental	$65	$72	$75	$78	$80	$90	$95	$100	$100
ACE Rent A Car	$68	$60	$62	$80	$92	$101	$110	$97	$97
Rent-A-Wreck	$102	$96	$90	$92	$78	$66	$48	$41	$36
Triangle Rent A Car	$35	$34	$33	$35	$37	$41	$45	$45	$45
Fox Rent A Car	N/A	N/A	N/A	$31	$50	$40	$62	$78	$81
Affordable / Sensible	N/A	N/A	N/A	$26	$33	$35	$36	$36	$36
Independent (3000+)	$1200	$1000	$785	$768	$750	$733	$635	$590	$540
Totals	$19400	$18200	$16430	$16460	$17640	$18,830	$20,413	$21,489	$21,879

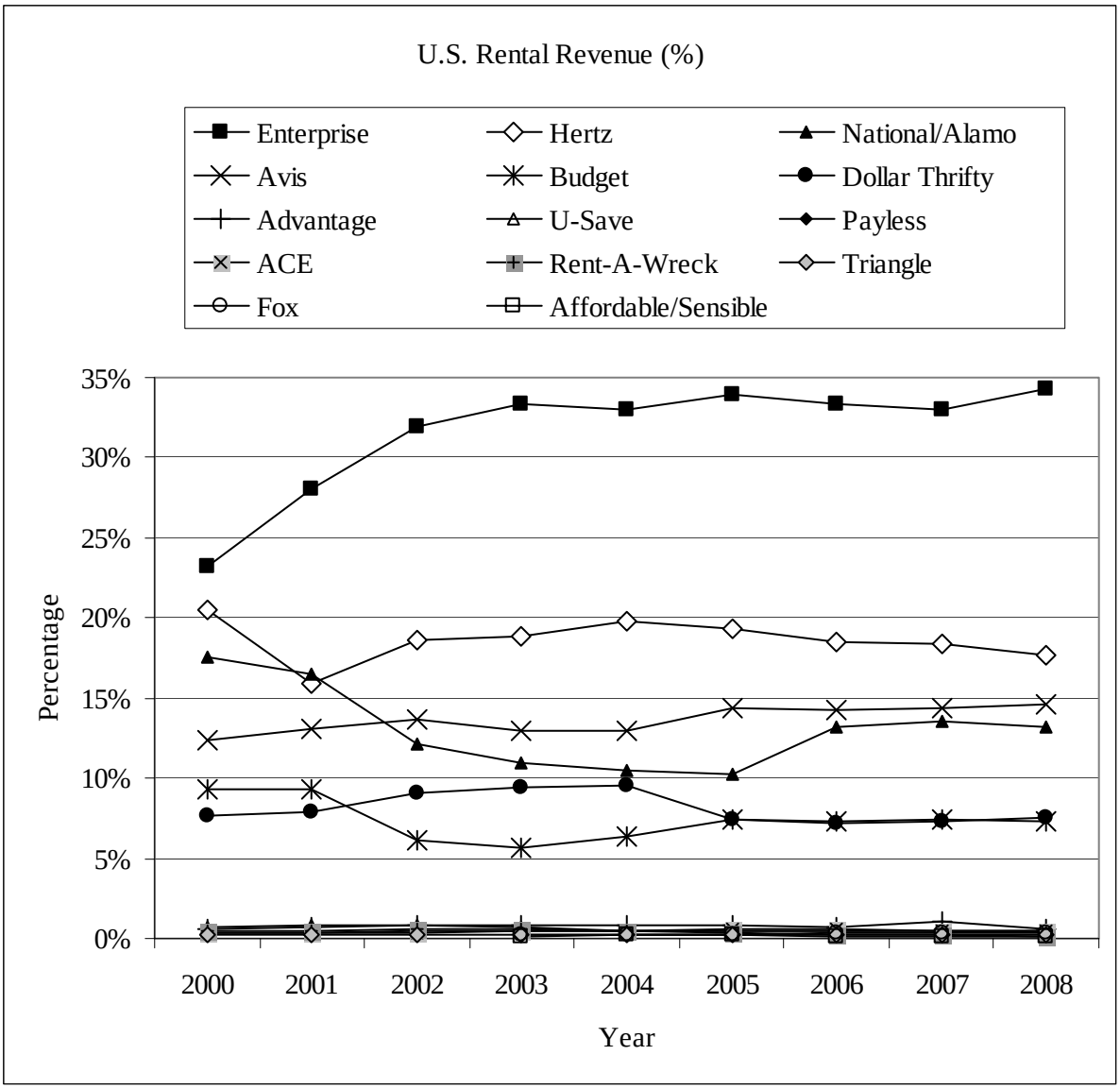

Figure 2.3. Market share of car rental revenue

2.3 Car Rental Software

The different functions of the software applications used by the car rental business are shown in Table 2.5. A tax function is commonly used for saving taxes such as 1031 exchange, which is a tax-deferred exchange. Wireless security is used for tracking the position and the location of the rented car. Rental management is used to manage the vehicle rental operations. The computerized reservation system handles internet booking. Accident management offers intelligent fleet solutions including accident reporting, vehicle repair, rental replacement, salvage, subrogation, and maintenance.

Table 2.5. Functions of car rental software applications

Software Function	Companies
Tax	Accruit LKE Solutions
Wireless Security	AirIQ
Rental Management	Bluebird, TSD, Enterprise Fleet Management
Computerized Reservation System	CRX, Sabre
Accident Management	Fleet Response, PAC

Rental management software is used mainly in fleet planning. TSD is the top car rental software and its major customers include Dollar Thrifty, Budget, and Avis. Bluebird also has many large clients including Rent-A-Wreck, Dollar Thrifty, and U-Save. Enterprise Rent-A-Car developed its own software, “Enterprise Fleet Management” to manage its fleet. “Enterprise Fleet Management” mainly includes web dashboard, maintenance management, insurance service, fuel savings, mileage report, full maintenance fees, vehicle acquisition,

state licenses, and remarketing. The functions of these rental management softwares, which are like database systems, are very similar to each other, and the software allows managers to manage their fleets based on the reports generated by the rental management software. Employees still need to assign the available cars to reserved customers. In addition, pool managers need to use forecasted demand from all locations to decide the car distribution plan within a pool for the next few days. These decisions are not evaluated and proved by accurate computations but instead require the manager's judgment and experience. Therefore, if a thorough fleet planning system covering long-term, mid-term, and short-term can be created and a complete plan of vehicle distribution can be rapidly and accurately obtained by algorithmic procedures, the manager's workloads will be largely reduced and misjudgments can be avoided.

CHAPTER 3

LITERATURE REVIEW

The literature pertaining to this dissertation is reviewed based on two main themes: (1) car rental problems (2) fleet planning problems. In car rental problems, three categories are discussed: revenue management, case studies, and fleet management. In fleet planning problems, long-term, mid-term, and short-term fleet planning are discussed individually.

3.1 Car Rental Problems

In a large car rental company, there are many aspects of the rental business that need to be addressed. The literature can be subdivided into three categories: revenue management, case study, and fleet management.

Revenue management is a management strategy that balances supply and demand in order to maximize profit. Tainiter (1964) introduced some stochastic inventory models to cope with the time fluctuations in the car rental business. Caseau and Kokeny (1998) proposed a set of practical benchmark instances to solve the overbooking problem. Kuyumcu and Garcia-Diaz (2000) designed a polyhedral graph theoretical approach utilizing a cutting plane and a branch-and-bound procedure to deal with the joint pricing and seat allocation

problem in the aircraft industry and obtained significant computer time-savings compared to integer programming commercial software.

Cooper (2002) used asymptotic properties of revenue management derived from a linear program to generate allocation policies and described counterintuitive behavior that could occur when allocations were updated during the booking process. Netessine *et al*. (2002) considered investing in a firm's capacity before the demand was known and upgrading customers to a higher level of service at no extra cost when the reserved car was not available. This short-term problem assigned capacity to customers when the demand was realized and was formulated as a two-stage single-period stochastic program. An efficient algorithm was developed to obtain the optimal capacities.

Bertsimas and Popescu (2003) designed a dynamic programming based algorithm, using an adaptive, non-additive bid price from a linear programming relaxation, to solve dynamic policies for allocating scarce inventory to stochastic demand for multi-fare classes. In addition, the proposed algorithm was extended to handle cancellation and no-show models by incorporating overbooking situations in the underlying linear programming formulation. Anderson *et al*. (2004) introduced a novel real options approach to revenue management in the car rental business. The proposed model generated acceptable prices and number of cars available for renting at a given price as a function of remaining time and inventory. This pricing and inventory model suggested current practices discount too deeply and too early in the booking cycle. However, this approach was limited to use in a single rental period and a single car class. Karaesmen and Ryzin (2004) formulated an overbooking problem with

multiple reservations and inventory classes as a two-period optimization problem and adopted a stochastic gradient algorithm to find the joint optimal overbooking level.

Hong *et al*. (2007) described the importance of forecasting monthly revenue per unit (RPU), which can provide a benchmarking index for annual pricing, and introduced three forecasting models, including the Holt-Winters' (HW) model, the seasonal Holt and Winters' Linear Exponential Smoothing (SHW) model, and the support vector regression (SVR) model. The numerical results revealed that SVRIA (support vector regression with immune algorithm) outperformed the other two models and provided a promising method of forecasting RPU. Cho and Rust (2008) proposed an econometric model to evaluate the automobile replacement policy adopted by a large car rental company. The simulation experiments revealed that the suggested alternative strategy, where cars were kept longer and the rental rates of old vehicles were cheaper, can produce an extra profit between 6% and 140%, depending on the car type. A summary of the research related to revenue management in the car rental business is presented in Table 3.1.

Table 3.1. Car rental literature on revenue management

	(Tainiter 1964)	(Caseau and Kokeny 1998)	(Kuyumcu and Garcia-Diaz 2000)	(Cooper 2002)	(Netessine, Dobson *et al*. 2002)	(Bertsimas and Popescu 2003)	(Anderson, Davison *et al*. 2004)	(Karaesmen and Ryzin 2004)	(Hong, Lai *et al*. 2007)	(Cho and Rust 2008)
Practical Field										
Airline			√	√	√	√	√	√		
Car Rental	√	√		√	√	√		√	√	√
Considerations										
Pricing	√						√		√	
Demand Forecasting			√							
Marginal Revenue			√						√	
Multi-Fare Classes	√	√	√	√	√	√		√		
Multi-Periods	√	√	√			√		√	√	
Cancellation						√		√		
No Show						√				
Overbooking		√				√		√		
Upgrade Policy					√					
Auto Replacement Policy										√
Modeling and Methodology										
Stochastic Process	√			√	√	√		√		
Linear Programming			√	√		√				
Dynamic Programming						√		√		
Integer Programming			√			√		√		
Decision Support System						√				
Real Options							√			
Cutting Plane			√							
Branch and Bound			√							
Support Vector Regression									√	
Evolutionary Algorithm									√	
Inventory Model		√								
Econometric Model										√

In case studies, Carroll and Grimes (1995) mainly introduced Hertz' decision support modules and the experiences in the car rental business. In fleet planning, the models Hertz adopted to build optimal overall fleet levels were a set of linear equations using past rental information, including the number of cancellations, and the estimated fleet utilization to produce aggregate fleet requirements, vehicle costs, and revenue per car per month. Spreadsheets were employed at the beginning of fleet planning. These tools permitted Hertz to better evaluate the trade-off between contribution and market share.

Geraghty and Johnson (1997) presented the crisis management of National Car Rental in 1993. National took steps to develop a revenue management program, which was a set of analytic models developed to manage capacity, pricing, and reservation, to avoid liquidation. National dramatically produced immediate results and returned National Car Rental to profitability in July 1993. New (2003) explored the impact of the multimedia work in Avis Europe on the field of operations management and offered an analysis of the experience of Avis Europe in developing a multimedia system for training frontline staff.

Lines *et al*. (2008) introduced the car rental study of a hydrogen strategy for transitioning from fleets to consumers in Orlando, FL. This study surveyed 435 consumers and the results indicated that half of all respondents were willing to pay more to rent a hydrogen car and that renting a pollution-free car was the most crucial deciding factor for this subset of customers. Then, Lines *et al*. pointed out the main barriers of building a hydrogen car fleet for a rental car company was the fleet purchase cost and proposed some practical solutions. A summary of case studies in the car rental business is presented in Table 3.2.

Table 3.2. Case studies in car rental business

	(Carroll and Grimes 1995)	(Geraghty and Johnson 1997)	(New 2003)	(Lines, Kuby *et al*. 2008)
Yield Management	√		√	
Revenue	√	√		
Fleet	√			
Operations			√	
Supply/Demand	√	√		
Green Fuel				√
Questionnaire				√
IT/DSS	√	√		
Hertz	√			
National		√		
Avis			√	

Fleet management in the car rental business covers many aspects, such as pool systems, fleet operations, empty flows, and others. Edelstein and Melnyk (1977) introduced the Pool Control System (PCS) constructed by Hertz Rent-A-Car. This system needed each city manager and each distribution manager to complete actual and projected data and then input the data to PCS in order to provide a detailed picture of the pool for the next seven days. Furth and Nash (1985) illustrated the benefit of a pooling operation in bus scheduling. This study indicated that a bus returning early can cover for a bus returning late. Dejax and Crainic (1987) surveyed and cataloged the empty flow problems and models in freight transportation. Inaba (2008) proposed a location inference model to infer the location of assets by using sparse RFID traceability information in the returnable transportation item rental service industry. A summary of fleet management in the car rental business is presented in Table 3.3.

Table 3.3. Fleet management in car rental business

	(Edelstein and Melnyk 1977)	(Furth and Nash 1984)	(Dejax and Crainic 1987)	(Inaba 2008)
Pool System	√	√		
Simulation				
Empty Flows			√	
RFID				√

3.2 Fleet Planning Problems

Fleet planning is normally separated into four groups based on transportation equipment---aircraft, freight, truck rental, and car rental. Aircraft fleet planning (Lohatepanont and Barnhart 2004; Li and Wang 2005) determines aircraft acquisition, flight routing, airline schedules, or the optimal utilization of aircrafts. Freight fleet includes rail freight (Bojovic 2002), truck freight (Hall 1999), air freight (Tyler 1986), and sea freight (Sambracos *et al.* 2004). The characteristics of truck rental (Wu *et al.* 2005; Martel 1990) and car rental (Pachon *et al.* 2003) are very much alike. They both must contend with fleet allocation and empty flow redistribution.

In the fleet planning literature, discussion of car rentals is very sparse. Pachon (2000) was basically the only paper that addresses long-term, mid-term, and short-term fleet planning in the car rental business. Pool segmentation was treated as a Minimum Spanning Tree problem. Pachon proposed a modified Kruskal's algorithm to cluster the pools in long-term planning. In mid-term planning, Pachon formulated a strategic fleet plan into a network flow problem. In addition, the set of complicated constraints was relaxed to a linear transshipment model with side constraints and was solved to optimality using optimization software. In short-term planning, Pachon solved a stochastic optimization model to determine daily deployment of the fleet.

All related literature has been divided into three sections for discussion. In Section 3.2.1, long-term planning is discussed mainly in the context of the classic location problems related to pool segmentation and hub selection. Section 3.2.2 addresses issues related to mid-term planning of car fleet operations, which solve the problem of inter-pool moves and asset

replacement. Demand allocation and empty flow redistribution, which is short-term car rental fleet planning, is presented in Section 3.2.3.

3.2.1 Long-term --- Pool Segmentation and Hub Selection

Pool segmentation clusters all locations into separate pools and selects one hub for each pool. Two classic location problems, the capacitated facility location problem with a single source constraint (Sridharan 1993) and the generalized assignment problem (Pentico 2007), are analogous to pool segmentation. A model that includes the cost of facility opening is called a capacitated facility location problem with a single source constraint. A single source constraint refers to the fact that each demand node can only be supplied by one facility. Models that do not include the cost of facility opening are generalized assignment problems. If a fixed number of districts are specified, the generalized assignment problem will simplify to a *p*-median problem (Mladenovic *et al.* 2007). The abundant literature offers a valuable direction for a solution methodology.

Sridharan (1993) proposed a Lagrangian relaxation heuristic to solve a capacitated facility location problem with a single source constraint. He relaxed the capacity constraint and solved this problem iteratively as a single plant location problem, a single source transportation problem with all plants open, and a knapsack problem. Syam (1997) designed a Lagrangian relaxation heuristic to solve a capacitated *p*-facility location problem and investigated some logistical issues that may be involved in managerial decision-making. Ronnqvist *et al.* (1999) reformulated the capacitated facility location problem with a single source constraint as a series of matching problems and introduced a repeated matching algorithm to solve until certain convergence criteria are satisfied. The numerical results

showed this approach to be much better than Pirkul's Lagrangian relaxation algorithm (1987). Holmberg *et al.* (1999) developed a class of heuristic algorithms for the capacitated facility location with a single source. This approach developed a repeated matching algorithm, incorporated into a Lagrangian heuristic and a branch-and-bound method based on a Lagrangian heuristic. The computational results showed the method to be very efficient. Klose (1999) described a linear programming-based algorithm for a two-stage capacitated facility location problem with a single source constraint; feasible solutions were obtained by utilizing linear programming and simple heuristics.

Ahuja *et al.* (2004) addressed a simple large scale neighborhood search algorithm for a capacitated facility location problem with a single source constraint. This approach continued from an initial solution with a sequence of facility moves and customer moves, and iterated until a local optimal solution is reached. Elhedhli and Goffin (2004) proposed an integrated algorithm based on an interior-point cutting-plane method within a branch-and-price scheme, which included decomposition techniques and a branch-and-bound approach. The overall approach was implemented for a capacitated facility location problem with a single source constraint and proved much more effective than Kelly's cutting-plane method (1960). Correia and Captivo (2006) presented a Lagrangian algorithm, enchanced by Tabu Search or local search to obtain feasible solutions. Test problems were randomly generated and showed this method able to obtain satisfactory solutions.

Chen and Ting (2006) suggested two ant colony systems to construct a heuristic procedure. One ant colony was used to select the opening facilities while the other one was utilized to allocate customers to each opening facility. Osman and Ahmadi (2007) indicated

that, in comparison with other known approaches in the literature, a guided construction search, based on a periodic local search procedure or a greedy adaptive search procedure, can obtain extremely good solutions for a capacitated *p*-median problem with a single source constraint.

Besides the capacitated facility location problem and the generalized assignment problem, Pachon (2000) simply considered the distance and formulated pool segmentation as a minimum spanning tree problem. A modified Kruskal's algorithm was proposed to cluster the pools. In this study (Wu, 2009), the problem of pool segmentation and hub selection is formulated as a capacitated facility location problem with a single source constraint. The distance cost, the hub opening cost, and the yearly demand are considered and car upgrade policy is addressed along with the constraint of pool capacity. A clustering-based iterative algorithm is presented to find a suitable solution quickly. A summary of the research related to pool segmentation is presented in Table 3.4.

Table 3.4. Literature in pool segmentation and hub selection

	(Sridharan 1993)	(Syam 1997)	(Ronnqvist *et al*. 1999)	(Holmberg *et al* 1999)	(Klose 1999)	(Ahuja, Orlin *et al*. 2004)	(Elhedhli and Goffin 2004)	(Correia and Captivo 2006)	(Osman and Ahmadi 2007)	(Pachon 2000)	**(Wu 2009)**
Problem Type											
Capacitated Facility Location	√	√	√	√	√	√	√	√			√
Capacitated Median Problem									√		
Minimum Spanning Tree										√	
Single Source Constraint	√		√	√	√	√	√	√	√		√
# of hubs is fixed						√		√	√		
Two Echelons					√						
Less than P Hubs		√									
Less than m nodes in a pool										√	
Car Rental Problem										√	√
Car Upgrade Policy											√
Solution Methodology											
Lagrangian Relaxation	√	√		√	√		√	√			
Metaheuristic					√	√		√	√		
Tabu Search								√			
Multi-Exchang						√					
Ant Colony											
Guided Construction									√		
LP-based heuristic					√						
Graph Algorithm			√	√							
Repeated Matching			√	√							
Clustering Algorithm											√
Kruskal's Algorithm										√	
Prim's Algorithm											√
Branch-and-Bound				√			√				
Cutting-Plane Method							√				
Column Generation							√				
Interior-Point Method							√				
Enumeration											√
Optimization											
Optimal Solution							√				
Approximation Solution	√	√	√	√	√	√		√	√	√	√
Numerical Example(max)											
Location	50	400	150	200	50	200	10	500	150	27	6,000
Problem Size (unit: 1,000)	2.5	160	4.5	6	250	6	0.6	500	23		3,601
Number of Problems	37	24	55	71	120	71	12	22	40	1	342

3.2.2 Mid-term--- Inter-pool Moves and Asset Replacement

In mid-term planning, Pachon (2000) formulated a strategic fleet plan into a network flow problem. In addition, the set of complicated constraints was relaxed to a linear transshipment model with side constraints and was solved to optimality using optimization software. The research proposed in this dissertation (Wu 2009) considers seasonal inter-pool moves and asset replacement and formulates a network flow model. A best-improvement descent local search, exploiting the structure of better neighbors, is proposed and validated. A summary of the research related to inter-pool moves and asset replacement is presented in Table 3.5.

Table 3.5. Literature in inter-pool moves and asset replacement

	(Pachon 2000)	**(Wu 2009)**
Modeling Approach		
Network Flow		√
Linear Programming	√	
Solution Methodology		
Neighborhood Search		√
LP Solver	√	
Considerations		
Inter-pool Moves	√	√
Asset Procurements and Sales		√
Different Car Types	√	√
Upgrade Car Type	√	√
Service Level		√
Leasing Contract	√	
Numerical Example (max)		
# of Pools	100	200
# of Car Types	5	8
# of Time Periods (season)	12	12
# of Car Ages (season)	12	12
Objective		
Max Profit		
Min Cost	√	√

3.2.3 Short-term --- Demand Allocation and Empty Flow Redistribution

Couillard and Martel (1990) introduced a mid-term seasonal stochastic model covering purchase, replacement, sale, and car rental. Additionally, an efficient algorithm and a decision support system were used to solve the model. Beaujou and Turnquist (1991) formulated an interactive fleet sizing and allocation model under dynamic and uncertain environments. An approximate network flow problem with a non-linear objective was solved by an interactive procedure using the Frank-Wolfe algorithm and was proved efficient enough to solve reasonably sized problems. Du and Hall (1997) developed decentralized stock control policies and an inventory approach in hub-and-spoke networks, and a decomposition method was utilized to find appropriate fleet size policies. Parikh (1977) assumed that all fleets provide a uniform level of service and adopted queueing theory to approximate the delay probability and the fleet size.

Pachon (2000) and Pachon *et al.* (2003) proposed the formulation of daily planning and decomposed the model into a fleet deployment model and a transportation model. Optimal conditions for both sub-problems and a heuristic to reduce the gap from optimality were introduced. Additionally, three extensions, including the cost of unsatisfied demand, service level, and a general price-demand function, were presented. Kochel *et al.* (2003) introduced a simulation method with a Genetic algorithm for the fleet sizing and allocation problems. Joborn *et al.* (2004) discussed empty freight car distribution and developed a Tabu heuristic to solve this time-dependent capacitated network model. Wu *et al.* (2005) suggested a linear programming model, which used a time-space network and considered demand allocation, empty truck redistribution, and asset procurements and sales, to determine optimal fleet size

and mix through a two-phase solution approach. Benders decomposition with a demand-shifting algorithm (Wu *et al.* 2003), was used to obtain feasible solutions in each subprogram in Phase I. Moreover, Phase II utilized the initial bounds and dual variables from Phase I to improve the solution convergence through the use of Lagrangian relaxation, resulting in effective solutions.

Fink and Reiners (2006) described short-term decisions in the car rental business to optimize fleet utilization and maintain a high service level. The authors proposed a system architecture of a decision support system which included a network model, a simulation model, and essential practical aspects, such as multi-period planning, a country-wide network, fleeting and defleeting, and car groups with partial substitutability. Simulation experiments showed that the network flow model led to a rental service level of 99.9% and an acceptable upgrade ratio of 16% in a seven-day period. Song and Earl (2008) addressed a two-depot optimal control policy for empty vehicle redistribution and fleet sizing problems, in which loaded vehicles arrival at depots and redistribution times for empty vehicles were uncertain. A novel stochastic model was introduced, and optimal threshold values and fleet size were derived. In this research (Wu 2009), a network flow model for daily planning in the same pool is developed. Empty flow redistribution, demand allocation, and car upgrade policy are considered. A first-improvement descent local search, exploiting the techniques of better neighbors, is developed. A summary of short-term fleet planning is presented in Table 3.6.

Table 3.6. Literature in demand allocation and empty flow redistribution

	(Parikh 1977)	(Beaujon and Turnquist 1991)	(Du and Hall 1997)	(Pachon 2000)	(Wu, Hartman *et al.* 2003)	(Pachon, Iakovou *et al.* 2003)	(Kochel, Kunze *et al* 2003)	(Joborn, Crainic *et al.* 2004)	(Wu, Hartman *et al.* 2005)	(Fink and Reiners 2006)	(Song and Earl 2008)	(Song and Carter 2008)	(Papier and Thonemann 2008)	**(Wu 2009)**
Practical Field														
Car Rental				√		√				√				√
Truck Rental			√		√				√		√			
Freight								√				√	√	
Modelling Approach														
Stochastic Process		√	√	√		√					√	√		
Linear Programming					√	√			√					
Decision Support System										√				
Network Flow								√		√				√
Queueing	√		√				√						√	
Inventory			√											
Solution Methodology														
Stochastic Optimization				√										
Decomposition Algorithm			√		√	√			√			√		
Lagrangian Relaxation									√					
Dynamic Programming											√	√		
Simulation							√			√	√			
Queueing	√		√				√						√	
Nonlinear Technique		√												
Metaheuristics / Neighborhood Search							√	√						√
Considerations														
Empty Flow Redistribution		√	√	√	√	√	√	√	√	√	√	√		√
Different Car Types				√				√	√	√			√	√
Upgrading Car type									√	√				√
Asset Procurements and Sales									√	√				
Stochastic Demand	√	√	√	√		√	√			√	√	√	√	
Unmet Demand		√	√	√			√						√	
Car Age									√	√				
Service Level														√
Numerical Example (max)														
# of Locations	1	10		2	10	6	5	50	30	500	3	100		21
# of Time Periods		7			30			5	60	14				21
# of Car Ages					2				3					
# of Rental Periods														21
# of Car Types				3	3				3	15			4	10
Objective														
Max Profit		√		√		√	√						√	
Min Cost					√			√	√	√	√			√
Min Fleet Size			√											
Balance the level of service	√													

CHAPTER 4

POOL SEGMENTATION AND HUB SELECTION

4.1 Problem Formulation

In long-term planning, it is assumed that all rental facilities are known. The distance between the facility and the opening hub is a known deterministic distance and the triangle inequality is satisfied. In addition, the yearly demand for each car type in each facility, the hub opening costs, the pool capacity limit, and the distance costs between any two nodes are known. All locations are split into several regions and one location is selected from each region to be the regional hub center. Moreover, the pool capacity limit is satisfied and the total cost of the transportation and the cost of opening the hubs is minimized. An example of networks before and after pool segmentation is shown in Figure 4.1 and Figure 4.2, respectively.

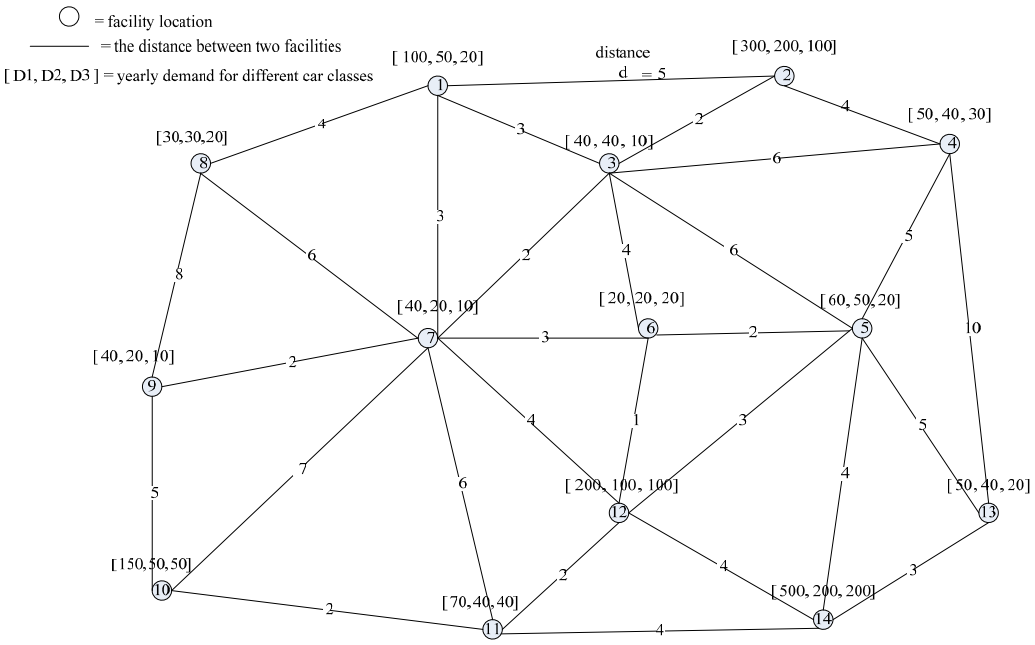

Figure 4.1. Network before pool segmentation

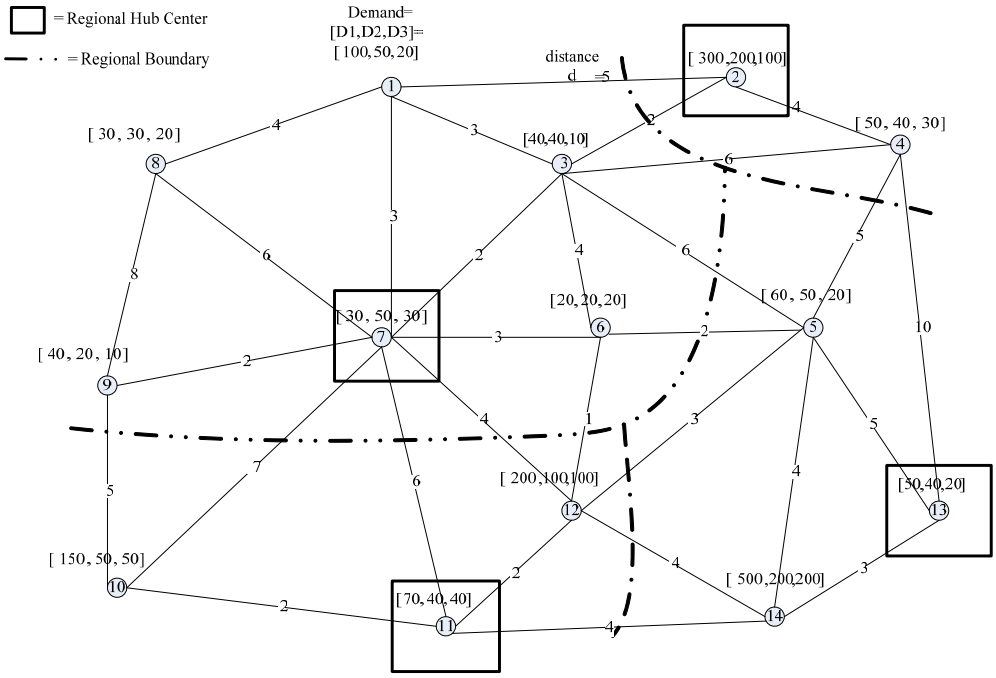

Figure 4.2. Network after pool segmentation

4.2 Mathematical Model

In this study, a model is formulated similarly to the capacitated facility location model with a single source constraint. However, an upgrade policy is included within the model. This pool segmentation and hub selection problem can be formulated as a binary integer linear programming.

Indices

i = location i
j = location j
k = car type k; the larger the value of k, the larger the car size

Parameters

d_{ij} = shortest distance cost between location i and location j
f_j = cost for opening hub at location j
D_i^k = annual forecasting demand for car type k at location i
u^k = upper bound for total demand of car type k at the same pool
l^k = lower bound for total demand of car type k at the same pool

Variables

$y_j = 1$ if location chosen to be a hub; otherwise, 0
$x_{ij} = 1$ if the demand at location i is assigned to hub j; otherwise, 0

The model can be expressed as:

$$\text{Min} \sum_k \sum_i \sum_j D_i^k d_{ij} x_{ij} + \sum_j f_j y_j \qquad (4.1)$$

subject to:

$$\sum_j x_{ij} = 1 \qquad \forall i \qquad (4.2)$$

$$x_{ij} \le y_j \qquad \forall i, j \qquad (4.3)$$

$$\sum_{k=k'}^{k_{\max}} l^k y_j \le \sum_{k=k'}^{k_{\max}} \sum_i D_i^k x_{ij} \le \sum_{k=k'}^{k_{\max}} u^k y_j \qquad \forall j, k', k' = \{1,2,...,k_{\max}\} \qquad (4.4)$$

$$x_{ij}, y_j \in \{0,1\} \qquad \forall i, j \qquad (4.5)$$

Objective (4.1) minimizes the total cost of the transportation costs plus the hub opening costs. Constraints (4.2) are called single source constraints. They ensure that each facility is assigned to exactly one hub. Constraints (4.3) address the fact that facility assignments are made only to open hubs. Constraints (4.4) indicate the capacity limit of regional hubs and the upgrade policy. The facility demand for a lower car type can be satisfied by the demand at the facility for a higher car type. For example, if a customer reserves a compact size car, but the compact size car is not available, then the car can be upgraded to an intermediate size, a standard size, or an even larger size car. That means that in car type k', the total pool demand for car type k' and higher needs to satisfy the total regional demand capacity of car type k' and higher. Constraints (4.5) are the integrality requirements.

4.3 Motivation

The mathematical model was verified and the solution form was observed utilizing LINGO software. Figure 4.3 is an example of an optimal solution with 60 facility locations. Pools group the nearby facilities together, which is analogous to clustering tribes with the smallest distance cost.

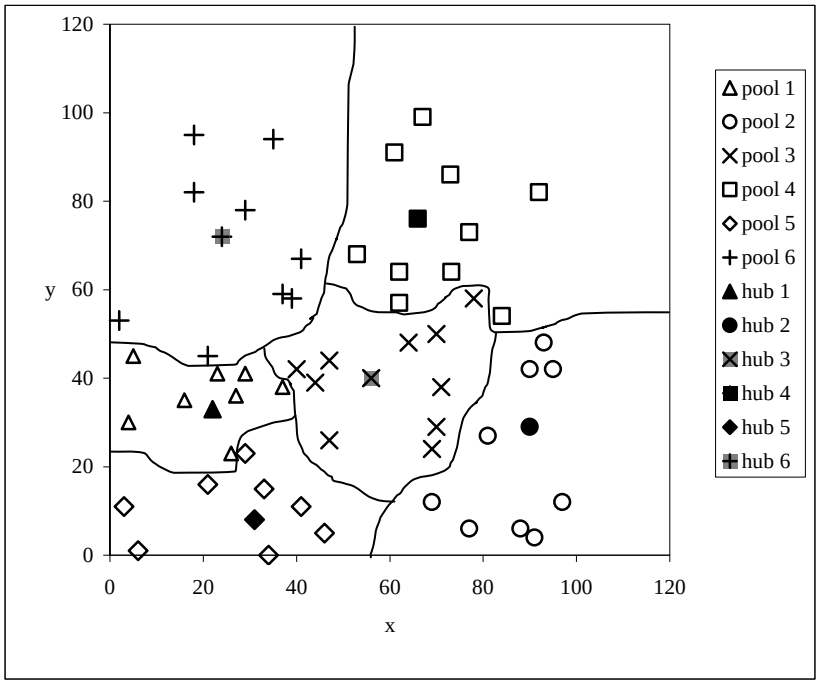

Figure 4.3. Example of an optimal solution with 60 facility locations

Observe that in the objective function $\sum_k \sum_i \sum_j D_i^k d_{ij} x_{ij} + \sum_j f_j y_j$ each facility incurs only one kind of cost. If a facility is not chosen to be the hub, then it will incur the distance cost. If a facility is chosen to be the hub, then the hub opening cost will be included. The distance cost is the product of the distance times the demand ($D \times d$) and the hub facility cost equals f. Figure 4.4 shows the solution form of the total cost incurred. A straight line

indicates that the distance cost is incurred and the black circle represents that the hub cost is incurred.

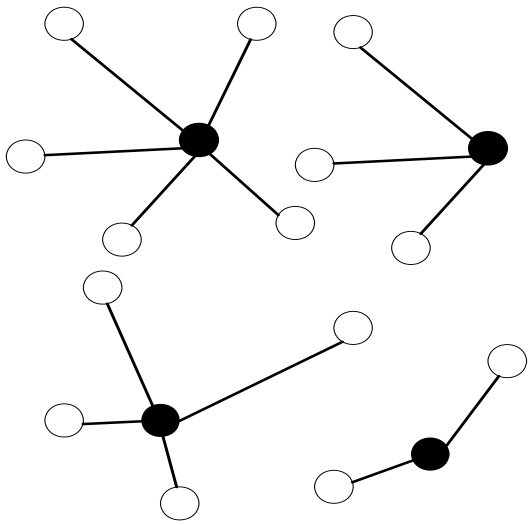

Figure 4.4. The solution form of the total cost incurred

Because the demands of each node are known, it is easier to evaluate the total cost if the hub facility cost can be adjusted to the demand description. For example, suppose that there are 7 nodes with 2 hubs. Nodes 1, 2, and 3 belong to pool 1, with node 2 the hub, and nodes 4, 5, 6, and 7 are in pool 2, with node 6 the hub. Therefore, the total cost will be $(\sum_k D_1^k d_{12} + f_2 + \sum_k D_3^k d_{32}) + (\sum_k D_4^k d_{46} + \sum_k D_5^k d_{56} + f_6 + \sum_k D_7^k d_{76})$, where k represents the car type. To more conveniently represent the hub facility cost for the algorithmic procedure, let $f_j = \sum_k D_j^k r_j$, where r_j represents the unit demand cost of hub j. Hence, the total cost $(\sum_k D_1^k d_{12} + f_2 + \sum_k D_3^k d_{32}) + (\sum_k D_4^k d_{46} + \sum_k D_5^k d_{56} + f_6 + \sum_k D_7^k d_{76})$ will become $(\sum_k D_1^k d_{12} + \sum_k D_2^k r_2 + \sum_k D_3^k d_{32}) + (\sum_k D_4^k d_{46} + \sum_k D_5^k d_{56} + \sum_k D_6^k r_6 + \sum_k D_7^k d_{47})$.

Let $h(j;i_1,i_2,...,i_r)=\dfrac{\sum_k D_j^k r_j+\sum_r\sum_k D_{i_r}^k d_{i_r j}}{\sum_k D_j^k+\sum_r\sum_k D_{i_r}^k}=\dfrac{f_j+\sum_r\sum_k D_{i_r}^k d_{i_r j}}{\sum_k D_j^k+\sum_r\sum_k D_{i_r}^k}$, where $h(j;i_1,i_2,...,i_r)$ represents the unit demand cost of pool j, including hub j and nodes $i_1,i_2,...,i_r$. The clustering algorithm tries to make the unit demand cost in each pool as small as possible. Hence, in pool 1, the unit demand cost is $\dfrac{\sum_k D_1^k d_{12}+f_2+\sum_k D_3^k d_{32}}{\sum_k (D_1^k+D_2^k+D_3^k)}$, and $\dfrac{\sum_k D_4^k d_{46}+\sum_k D_5^k d_{56}+f_6+\sum_k D_7^k d_{76}}{\sum_k (D_4^k+D_5^k+D_6^k+D_7^k)}$ for pool 2. This resulting term is called the unit demand cost of the pool.

Because of all the known demands and the idea of local optimality, the smaller the distance, the lower the cost. However, in this formulation, it is unknown how many hubs are needed; only the capacity of the pool is known. Hence, in the beginning, the minimum unit demand cost of each possible hub is compared. One node is specified as the hub and other nodes are linked to this hub node to meet two requirements:

- Meet the lower bound of the capacity, without exceeding the upper bound
- Meet the minimal unit demand cost of this pool

Based on this concept, a very good initial solution can be found and the number of hubs is therefore close to or equivalent to the real optimal number of hubs. Table 4.1 offers a good validation for this rule. The clustering algorithm is described in detail in Section 4.4.

Table 4.1. Optimal solution vs. clustering algorithm

Example		1	2	3
Number of nodes		90	160	200
Number of car types		5	10	10
Clustering algorithm	Cost ($)	617,785.1	3,102,533.5	4,234,375
	Number of hubs	32	24	31
Optimal	Cost ($)	603,013.9	3,050,300	4,212,918
	Number of hubs	32	21	29
Solution gap (%)		2.45%	1.71%	0.51%

In addition, Example 1 is re-solved with 90 locations, but the number of hubs is fixed. That makes the problem solvable as a capacitated *p*-facility location problem. The optimal solutions for different numbers of hubs are shown in Figure 4.5 and built into a convex function. The clustering algorithm can approach the optimal number of hubs and the optimal solution as the black box shows in Figure 4.5. If the number of hubs is determined, then the whole problem will reduce to solving the capacitated *p*-facility location problem. Based on the characteristic of a convex function, only a small number of capacitated *p*-facility location problems need to be solved.

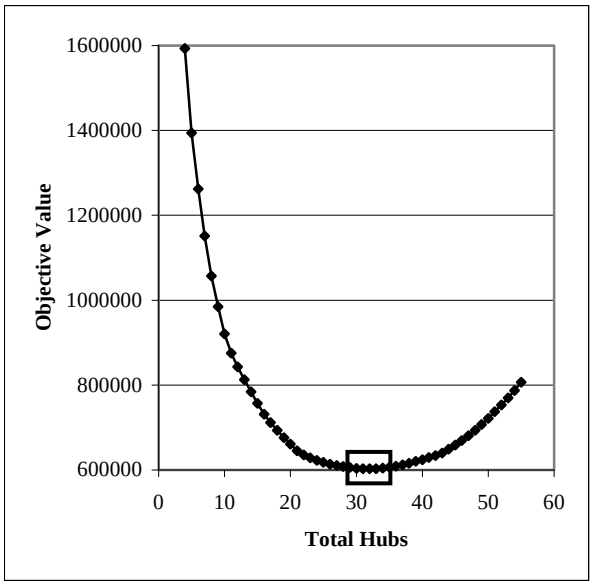

Figure 4.5. Optimal value vs. different numbers of hubs

In solving a capacitated *p*-facility location problem, an iterative procedure is adopted to address the process. Each pool region, obtained from the initial solution of the clustering algorithm, is fixed as presented in Figure 4.6 and each hub is easily re-optimized by an enumeration method. Once each hub is re-allocated, the locations of the hubs are fixed and all the other facility locations are re-located to these fixed hubs as shown in Figure 4.7. This becomes a multi-resource generalized assignment problem. Although a branch-and-bound algorithm is widely used in multi-resource generalized assignment problems (Park *et al.* 1998; Babkin *et al.* 1977; Fei *et al.* 2008; Nauss 2004; Haddadi and Ouzia 2004), it still cannot solve practical problems in polynomial time. Hence, a modified Prim's algorithm is implemented in order to obtain a near-optimal solution. Then these two steps are executed iteratively to obtain a near-optimal solution for the capacitated *p*-facility location problem.

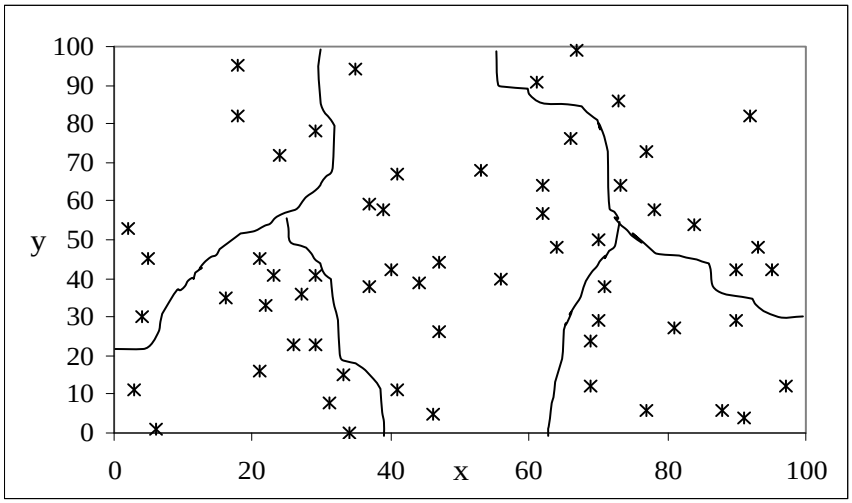

Figure 4.6. Fixing the pool region and re-selecting the hub

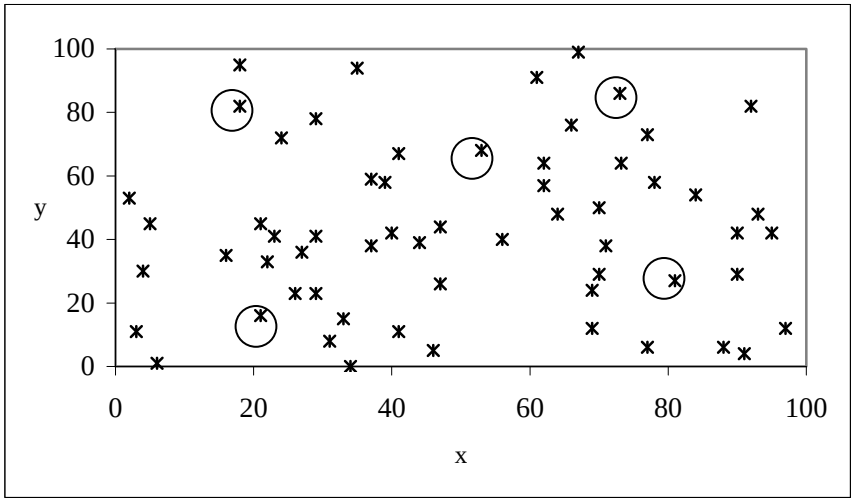

Figure 4.7. Fixing the hubs and re-shaping the pool regions

After obtaining a near-optimal solution for the capacitated *p*-facility location problem, the number of hubs is adjusted using this technique, by either adding or deleting one hub, and the iterative procedure is re-executed. A near-optimal solution for the whole problem can be obtained.

4.4 Algorithm Procedure

Based on the previous motivation in Section 4.3, a clustering-based iterative algorithm is introduced. This algorithm mainly includes three modules: a clustering algorithm, an enumeration method, and a modified Prim's algorithm. The whole rough process of the three phases of the algorithm is displayed in Figure 4.8.

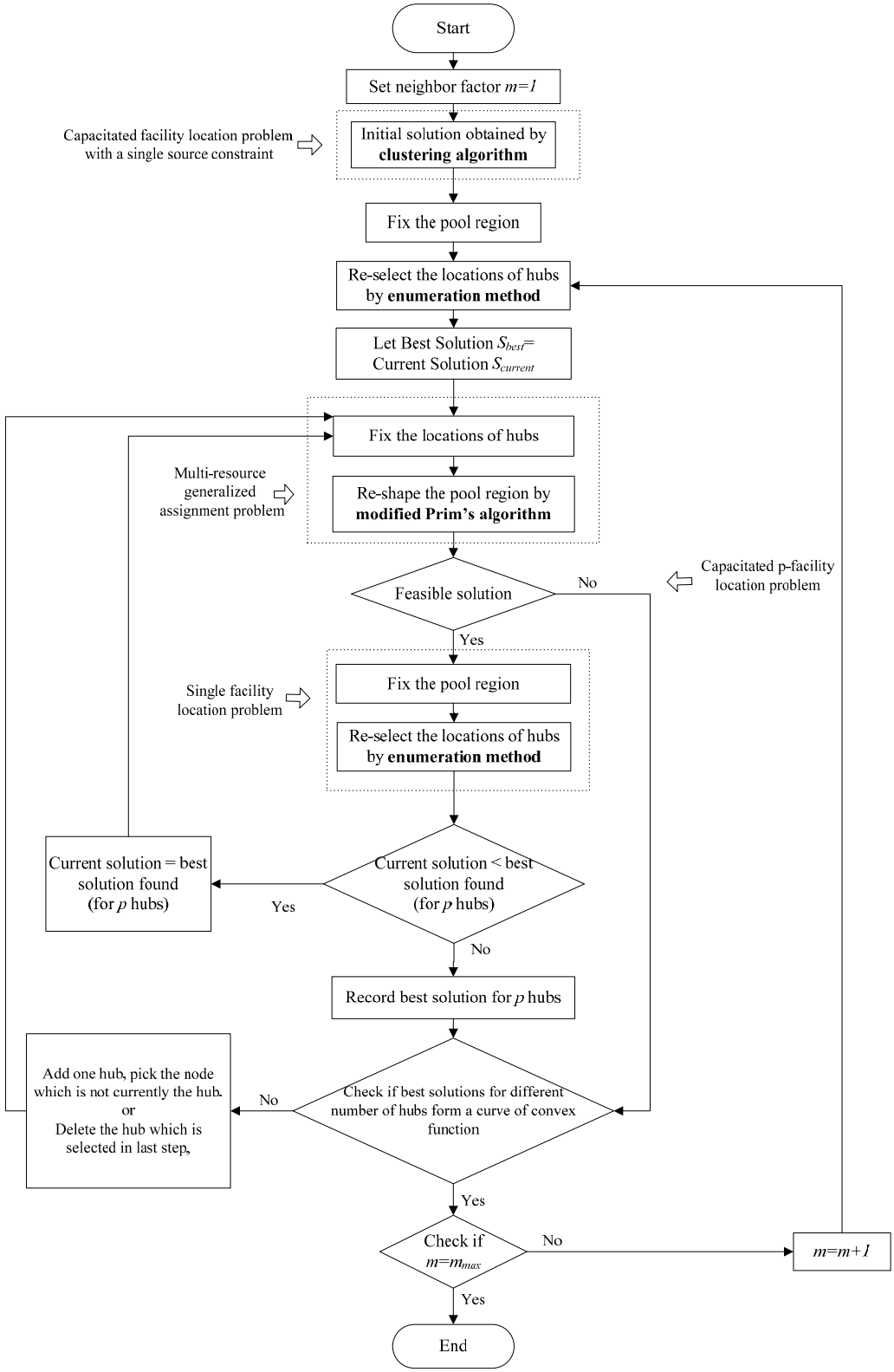

Figure 4.8. The clustering-based iterative algorithm

The processes of the three important modules are explained in detail below.

4.4.1 Clustering Algorithm

4.4.1.1 Neighbor Factor

The neighbor factor is an integer value and it is used to calculate and extend the list of all candidate pool regions. If the value of a neighbor factor is m, the neighbor index is calculated from 1 to *m*. If the neighbor index is *m*, that means that this candidate pool region selects its covered nodes from the m-th closest node and the previous $m-1$ closet nodes are not covered in this pool region. For example, if the value of a neighbor index is 3, that means that this candidate pool region selects its covered nodes from the third closest node and the previous two closet nodes are not covered in this pool region. Because a value of neighbor factor *m* covers the neighbor indexes from 1 to *m*, it can extend the list of all candidate pool regions and thereby increase the possibility of obtaining a good solution.

4.4.1.2 Tabu List

The Tabu List (*TL*) is a module of Tabu Search, proposed by Fred Glover.(1977). The Tabu List is a memory structure used to record past moves and avoid the formation of cycle moves. In the Tabu List, FIFO rules are usually applied. That means that the newest move is added to memory and the oldest move is removed from memory. The larger the Tabu List, the less the possibility of falling into a local optimum. However, more computer memory space is needed and computing time is significantly longer. Generally speaking, the value of the Tabu List doesn't have limits. It is usually decided by the characteristic of the problem.

4.4.1.3 Flow Chart

The rough procedure of the clustering algorithm is presented in Figure 4.9 and Figure 4.10.

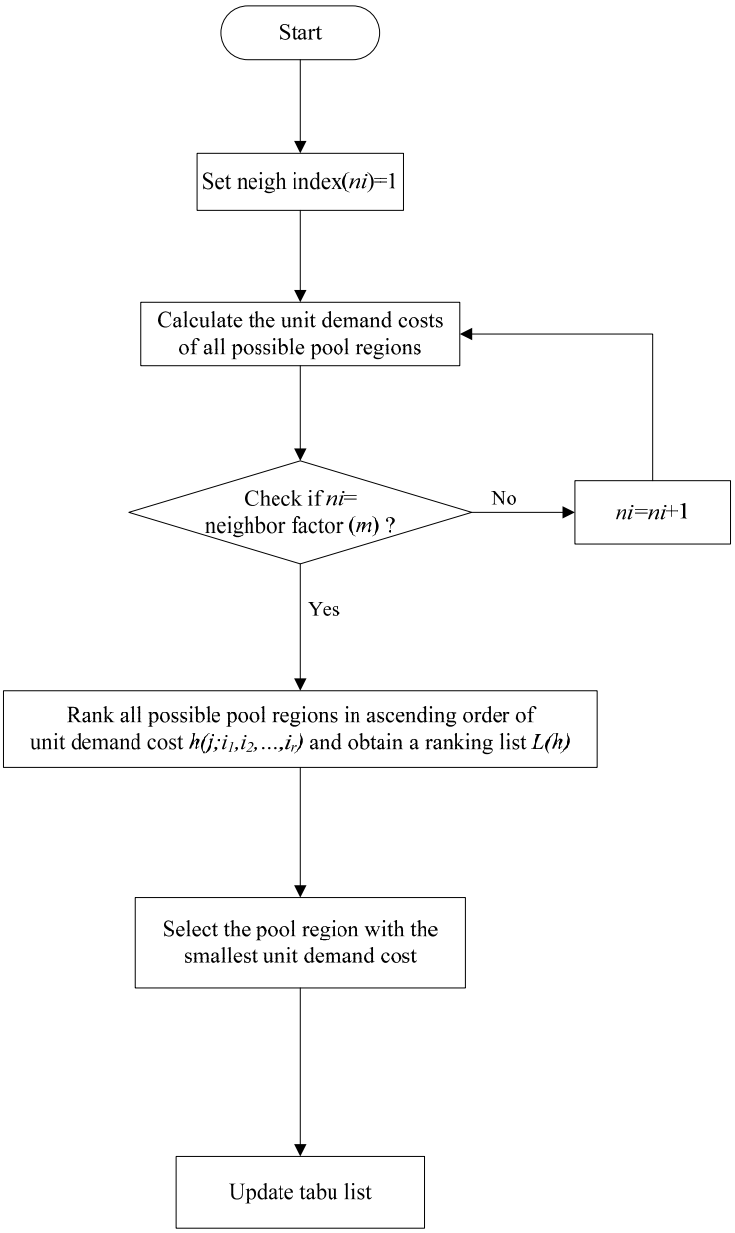

Figure 4.9. The clustering algorithm(1/2)

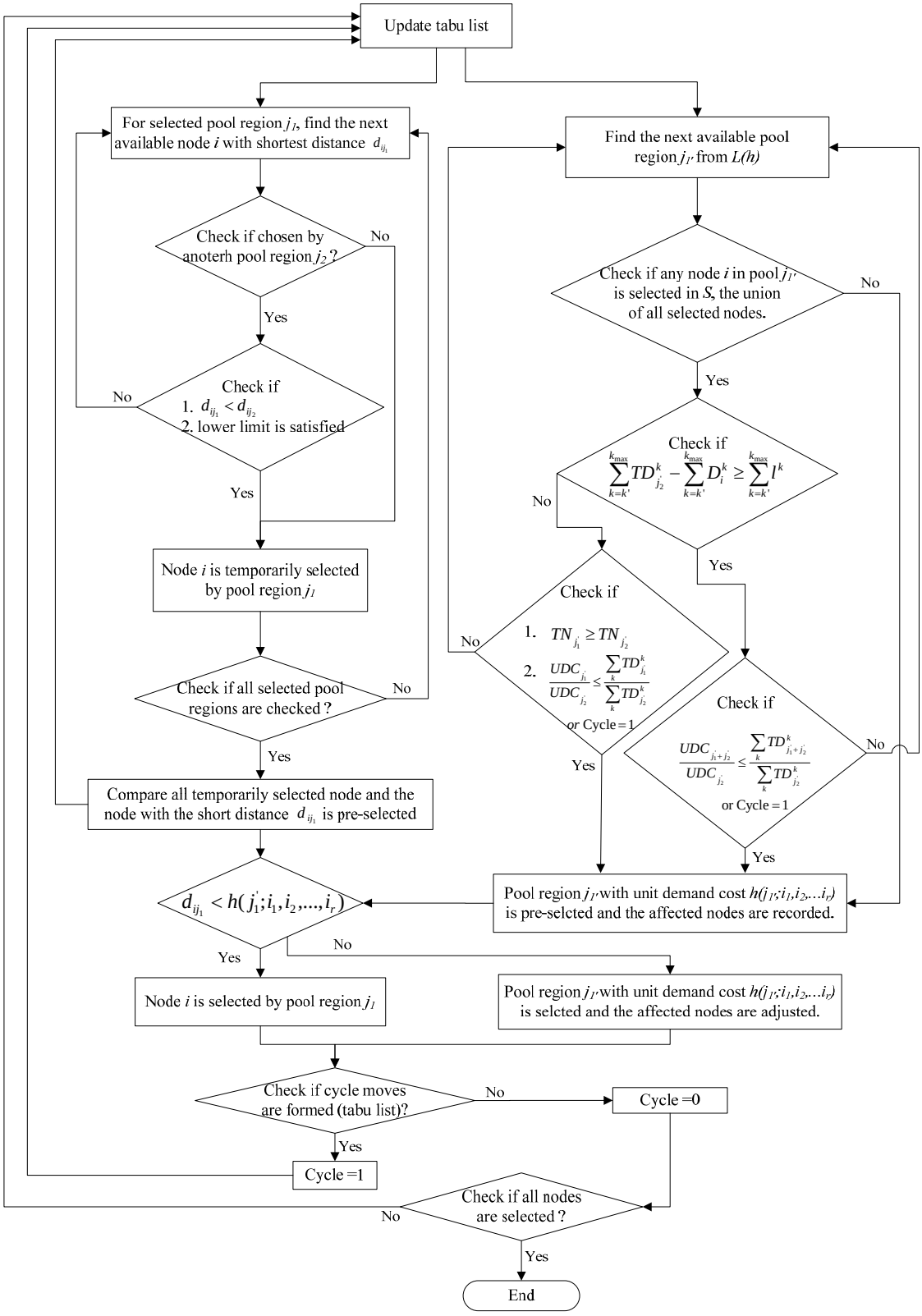

Figure 4.10. The clustering algorithm(2/2)

4.4.1.4 Detailed Procedure

Step 1: Set the neighbor index (*ni*) =1. For each candidate hub j, calculate the unit demand cost of hub j, $r_j = \frac{f_j}{\sum_k D_j^k}$, where k represents the car type. Consider the distance d_{ij} for all nodes $i \neq$ hub j. Rank these nodes i in ascending order of d_{ij} and obtain a list of ranked nodes $\{d_{i,m_1}, d_{i,m_2}, d_{i,m_3}, ..., d_{i,m_{j\max}}; d_{i,m_1} \le d_{i,m_2} \le d_{i,m_3} \le ... \le d_{i,m_{j\max}}\}$, denoted as L_{j_1}. Go to Step 3.

Step 2: If $ni \neq 1$, let $L_{j_{ni}} = L_{j_1} - \{d_{i,m_1}, d_{i,m_2}, ..., d_{i,m_{(ni-1)}}\}$. Go to Step 3.

Step 3: For each candidate hub j, if $r_j < d_{ij}$ for all $i \neq j$, go to Step 4; otherwise, go to Step 5. Continue with Step 3 to check all candidate hubs; then go to Step 6.

Step 4: Calculate the total demand of car type k at hub j, $TD_j^k(i_1, ..., i_r) = D_j^k + \sum_r D_{i_r}^k$. Check if $\sum_{k=k'}^{k_{\max}} TD_j^k(i_1, ..., i_r) \ge \sum_{k=k'}^{k_{\max}} l^k$, *for* all car type k'. If yes, record r_j and go to Step 3 to check other candidate hubs. If not, add nodes $i_r \in L_{j_{ni}}$ to pool j and re-calculate the unit demand cost $h(j; i_1, i_2, ..., i_r) = \frac{f_j + \sum_r \sum_k D_{i_r}^k d_{i_r j}}{\sum_k D_j^k + \sum_r \sum_k D_{i_r}^k}$ of pool j, which includes hub j and nodes $i_1, i_2, ..., i_r$ until $\sum_{k=k'}^{k_{\max}} TD_j^k(i_1, i_2, ..., i_r) \ge \sum_{k=k'}^{k_{\max}} l^k$, for all car types k'. Go to Step 3.

Step 5: Add nodes $i_r \in L_{j_{ni}}$ to pool j and re-calculate the unit demand cost $h(j; i_1, , i_2, ..., i_r) = \dfrac{f_j + \sum_r \sum_k D_{i_r}^k d_{i_r j}}{\sum_k D_j^k + \sum_r \sum_k D_{i_r}^k}$ until these two following conditions are satisfied. Go to Step 3.

(a) $\sum_{k=k'}^{k_{\max}} u^k \geq \sum_{k=k'}^{k_{\max}} TD_j^k(i_1, i_2, ..., i_r) \geq \sum_{k=k'}^{k_{\max}} l^k$, for all car types k'

(b) $h(j; i_1, i_2, ..., i_r) \geq h(j; i_1, i_2, ..., i_r, i_{r+1})$

Step 6: Check if *ni*=neighbor factor(*m*). If not, let *ni=ni+1* and go back to Step 2; otherwise, go to Step 7.

Step 7: Based on Step 1~Step 6, one can obtain a stream of nodes with the sequence for each candidate hub. The pool j_1 with the smallest unit demand cost $h(j_1; i_1, i_2, ..., i_r)$ is selected and recorded in the Tabu List. Let cycle=0. Go to Step 8.

Step 8: For every selected pool j_1, find the node $i \in L_{j_1}$ with the shortest distance d_{ij_1}. If node i has been added to another pool region j_2, the distance d_{ij_1} is compared to d_{ij_2}. If $d_{ij_1} < d_{ij_2}$ and $\sum_{k=k'}^{k_{\max}} TD_{j_2}^k(i_1, i_2, ..., i_r) - \sum_{k=k'}^{k_{\max}} D_i^k \geq \sum_{k=k'}^{k_{\max}} l^k$ for all car type k', node i is added to pool j_1, i.e. P_{j_1} ; otherwise, find the next node $i' \in L_{j_1}$ until all nodes are evaluated. Then the shortest distance d_{ij_1} of pool j_1 is selected. That means that node i is added to pool j_1. If more than one node has the smallest distance, the node with larger demand is picked. Go to Step 9.

Step 9: For the unselected candidate hubs recorded in Step 7, find the next pool $j_1^{'}$, where nodes $i_1, i_2, ..., i_r$ are added to $j_1^{'}$, with the smallest unit demand cost. Four possible cases exist:

(a) If all nodes $i \in$ pool $j_1^{'}, P_{j_1^{'}}$, and $i \notin S$, where S is the union of all selected nodes, pool $j_1^{'}$ is selected and its unit demand cost $h(j_1^{'}; i_1, i_2, ..., i_r)$ is recorded. Go to Step 10.

(b) If any node $i \in P_{j_1^{'}}$, $i \in P_{j_2^{'}}$, $i \in S$, and $\sum_{k=k'}^{k_{\max}} TD_{j_2^{'}}^k (i_1^{'}, i_2^{'}, ..., i_{r'}^{'}) - \sum_{k=k'}^{k_{\max}} D_i^k \geq \sum_{k=k'}^{k_{\max}} l^k$ *for* all k', calculate the unit demand cost and the total demand of pool $j_1^{'}$ plus $j_2^{'}$. The unit demand cost $UDC_{j_1^{'}+j_2^{'}}$ of pool $j_1^{'}$ plus $j_2^{'}$ is $\frac{h(j_1^{'}; i_1, ..., i_r) \times \sum_k TD_{j_1^{'}}^k + h(j_2^{'}; i'_1, ..., i'_{r'}) \times \sum_k TD_{j_2^{'}}^k - \sum_k D_i^k d_{ij_2^{'}}}{\sum_k TD_{j_1^{'}}^k + \sum_k TD_{j_2^{'}}^k - \sum_k D_i^k}$. The total demand $\sum_k TD_{j_1^{'}+j_2^{'}}^k$ of pool $j_1^{'}$ plus $j_2^{'}$ is $\sum_k TD_{j_1^{'}}^k + \sum_k TD_{j_2^{'}}^k - \sum_k D_i^k$. If $\frac{UDC_{j_1^{'}+j_2^{'}}}{UDC_{j_2^{'}}} \leq \frac{\sum_k TD_{j_1^{'}+j_2^{'}}^k}{\sum_k TD_{j_2^{'}}^k}$ or cycle=1, pool $j_1^{'}$ is selected and the affected nodes are recorded. Go to Step 10.

(c) If any node $i \in P_{j_1^{'}}$, $i \in P_{j_2^{'}}$, $i \in S$, and $\sum_{k=k'}^{k_{\max}} TD_{j_2^{'}}^k (i_1^{'}, i_2^{'}, ..., i_{r'}^{'}) - \sum_{k=k'}^{k_{\max}} D_i^k < \sum_{k=k'}^{k_{\max}} l^k$ *for* any k', calculate the total number $TN_{j_1^{'}}$ of nodes at pool $j_1^{'}$ and the total

number $TN_{j_2'}$ of nodes at pool j_2'. If (1) $TN_{j_1'} \geq TN_{j_2'}$ and (2) $\frac{UDC_{j_1'}}{UDC_{j_2'}} \leq \frac{\sum_k TD_{j_1'}^k}{\sum_k TD_{j_2'}^k}$ or

cycle=1, pool j_1' is selected and the affected nodes are recorded. Go to Step 10.

(d) If none of the three above cases exist, find the next hub until all candidate hubs are evaluated.

Step 10: Compare d_{ij_1} in Step 8 to $h(j_1';i_1,i_2,...,i_r)$ in Step 9. If $d_{ij_1} < h(j_1';i_1,i_2,...,i_r)$, node i is added to pool j_1. Otherwise, pool j_1' is selected. That means that nodes $i_1,i_2,...,i_r$ are added to hub j_1'. Check if a cycle is formed in the Tabu List. If yes, cycle =1. Update the Tabu List and Go to Step 8. Otherwise, cycle =0 and check if all nodes have been selected. If yes, this algorithm is complete; otherwise, update the Tabu List and go to Step 8.

4.4.1.5 Example

For example, the 14 nodes, as indicated in Figure 4.11, have a distance matrix, as displayed in Table 4.2. The neighbor factor is assumed to be 1.

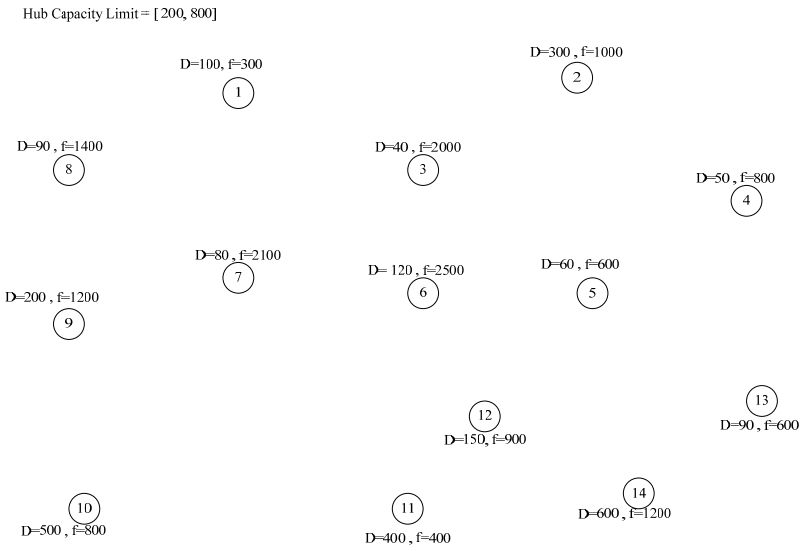

Figure 4.11. Example of 14 locations with their demands and hub opening costs

Table 4.2. Distance matrix for the 14 nodes

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1		5	3	9	8	6	3	4	5	10	9	7	13	11
2	5		2	4	8	6	4	9	6	11	9	7	13	11
3	3	2		6	6	4	2	7	4	9	7	5	11	9
4	9	4	6		5	7	8	13	10	12	10	8	10	9
5	8	8	6	5		2	5	11	7	7	5	3	5	4
6	6	6	4	7	2		3	9	5	5	3	1	7	5
7	3	4	2	8	5	3		6	2	7	6	4	10	8
8	4	9	7	13	11	9	6		8	13	12	10	16	14
9	5	6	4	10	7	5	2	8		5	7	6	12	10
10	10	11	9	12	7	5	7	13	5		2	4	9	6
11	9	9	7	10	5	3	6	12	7	2		2	7	4
12	7	7	5	8	3	1	4	10	6	4	2		7	4
13	13	13	11	10	5	7	10	16	12	9	7	7		3
14	11	11	9	9	4	5	8	14	10	6	4	4	3	

[I].

The demand bound for the pool region = (200, 800). Calculate all possible hubs:

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 1	100	0	300	300	100	3	No	
Node 7	80	3	240	540	180	3	No	
Node 3	40	3	120	660	220	**3**	**Yes**	**Yes**
Node 8		4 (4>3)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 2	300	0	1000	1000	300	**3.33**	Yes	
Node 3	40	2	80	1080	340	**3.176**	**Yes**	**Yes**
Node 7		4 (4>3.176)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 3	40	0	2000	2000	40	50	No	
Node 2	300	2	600	2600	340	**7.647**	Yes	
Node 7	80	2	160	2760	420	**6.571**	Yes	
Node 1	100	3	300	3060	520	**5.885**	Yes	
Node 9	200	4	800	3860	720	**5.361**	**Yes**	**Yes**
Node 6	120	4			840 (840>800)		No	
Node 12	150	5			870 (870>800)		No	
Node 5	60	6 (6>5.361)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 4	50	0	800	800	50	16	No	
Node 2	300	4	1200	2000	350	**5.714**	Yes	
Node 5	60	5	300	2300	410	**5.610**	**Yes**	**Yes**
Node 3		6 (6>5.610)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 5	60	0	600	600	60	10	No	
Node 6	120	2	240	840	180	4.667	No	
Node 12	150	3	450	1290	330	**3.909**	**Yes**	**Yes**
Node 14		4 (4>3.909)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 6	120	0	2500	2500	120	20.83	No	
Node 12	150	1	150	2650	270	**9.815**	Yes	
Node 5	60	2	120	2770	330	**8.394**	Yes	
Node 11	400	3	1200	3970	730	**5.438**	Yes	
Node 7	80	3			810 (810>800)	8.394	No	
Node 3	40	4	160	4130	770	**5.364**	**Yes**	**Yes**
Node 14	600	5			1370 (1370>800)		No	
Node 10	500	5			1270 (1270>800)		No	
Node 9	200	5			970 (970>800)		No	
Node 2		6 (6>5.364)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 7	80	0	2100	2100	80	26.25	No	
Node 9	200	2	400	2500	280	**8.929**	Yes	
Node 3	40	2	80	2580	320	**8.063**	Yes	
Node 6	120	3	360	2940	440	**6.682**	Yes	
Node 1	100	3	300	3240	540	**6**	Yes	
Node 2	300	4	1200	4440	840 (840>800)		No	
Node 12	150	4	600	3840	690	**5.565**	Yes	
Node 5	60	5	300	4140	750	**5.52**	**Yes**	**Yes**
Node 11	400	6 (6>5.52)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 8	90	0	1400	1400	90	15.556	No	
Node 1	100	4	400	1800	190	9.474	No	
Node 7	80	6	480	2280	270	**8.444**	Yes	
Node 3	40	7	280	2560	310	**8.258**	Yes	
Node 9	200	8	1600	4160	510	**8.157**	**Yes**	**Yes**
Node 2		9 (9>8.157)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 9	200	0	1200	1200	200	**6**	Yes	
Node 7	80	2	160	1360	280	**4.857**	Yes	
Node 3	40	4	160	1520	320	**4.75**	**Yes**	**Yes**
Node 10		5 (5> 4.75)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 10	500	0	800	800	500	**1.6**	**Yes**	**Yes**
Node 11		2 (2>1.6)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 11	400	0	400	400	400	**1**	**Yes**	**Yes**
Node 10		2 (2>1)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 12	150	0	900	900	150	6	No	
Node 6	120	1	120	1020	270	**3.778**	Yes	
Node 11	400	2	800	1820	670	**2.716**	**Yes**	**Yes**
Node 5		3 (3>2.716)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 13	90	0	600	600	90	6.667	No	
Node 14	600	3	1800	2400	690	**3.478**	**Yes**	**Yes**
Node 5		5 (5>3.478)						No

	Demand	Distance	Cost	Total cost	Total demand	Unit demand cost	Meet the capacity	Min
Hub 14	600	0	1200	1200	600	**2**	**Yes**	**Yes**
Node 13		3 (3>2)						No

Once all possible hubs are calculated, a ranking list of all candidate pool regions is obtained as shown in Table 4.3.

Table 4.3. A ranking list of all candidate pool regions

Ranking	Unit demand cost	Hub	Connecting Nodes	Total Demand
1	1	11		400
2	1.6	10		500
3	2	14		600
4	2.716	12	6,11	670
5	3	1	7,3	220
6	3.176	2	3	340
7	3.33	2		300
8	3.478	13	14	690
9	3.778	12	6	270
10	3.909	5	6,12	330
11	4.75	9	7,3	320
12	4.857	9	7	280
13	5.361	3	2,7,1,9	720
14	5.364	6	12,5,11,3	770
15	5.438	6	12,5,11	730
16	5.52	7	9,3,6,1,12,5	750
17	5.565	7	9,3,6,1,12	690
18	5.61	4	2,5	410
19	5.714	4	2	350
20	5.885	3	2,7,1	520
21	6	7	9,3,6,1	540
22	6	9		200
23	6.571	3	2,7	420
24	6.682	7	9,3,6	440
25	7.647	3	2	340
26	8.063	7	9,3	320
27	8.157	8	1,7,3,9	510
28	8.258	8	1,7,3	310
29	8.394	6	12,5	330
30	8.444	8	1,7	270
31	8.929	7	9	280
32	9.815	6	12	270

Select the hub with the smallest unit demand cost (r_{11}=1). Node 11 is chosen to be a hub as shown in Figure 4.12.

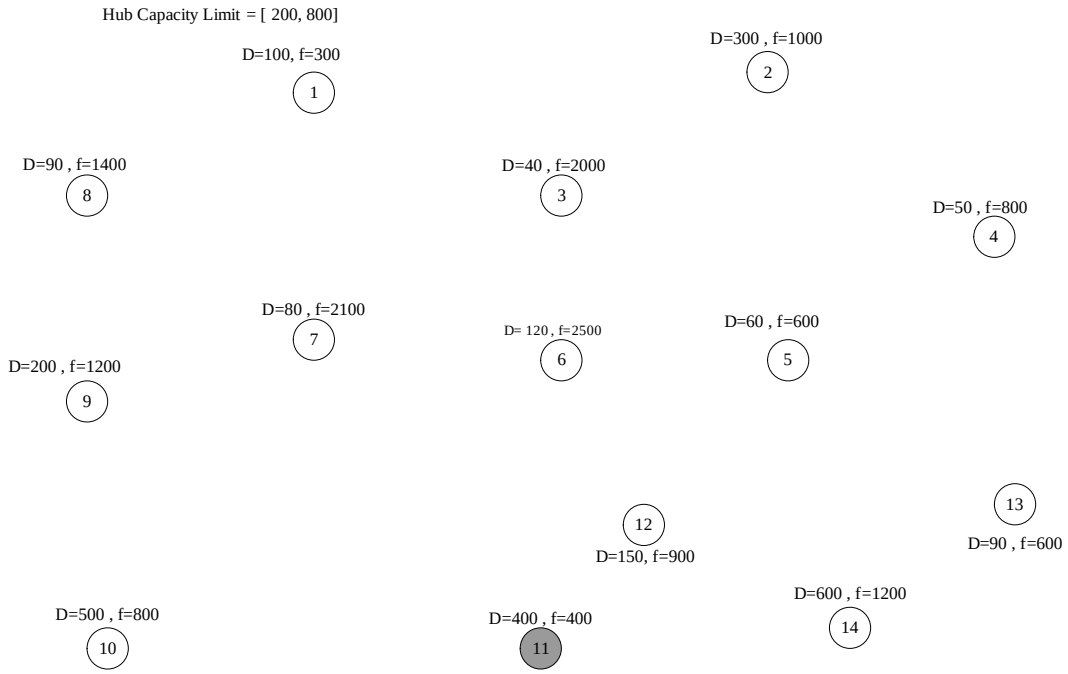

Figure 4.12. Node 11 is selected to be a hub

[II]

Compare the two possible choices and choose the smallest.

(1) Add one node with the shortest distance to the selected hub. However, the demand capacity for the pool still needs to be satisfied. Otherwise, check the next available node.

(2) Check other possible hubs from Table 4.3. Choose the next available pool region with the smallest unit demand cost. If a node has not been selected, check the unit demand cost.

For (1), the shortest distance for hub 11 is $d_{11,12} = 2$ and as a result, node 12 is selected for possible addition to the hub 11 pool, P_{11}. For (2), the smallest unit demand cost is $r_{10} = 1.6$ and hub 10 is selected for possible additional node. Since $r_{10} < d_{11,12}$, node 10 is selected to be a hub, as shown in Figure 4.13.

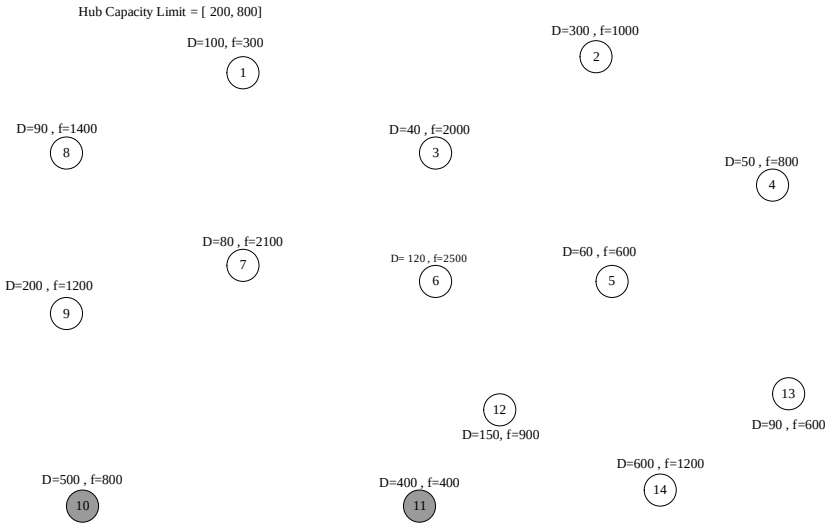

Figure 4.13. Node 10 is selected to be a hub

[III]

This procedure is the same as [II]. Two possible choices exist.

For (1), the possible selection for the shortest distance for hub 11 is $d_{11,12} = 2$, and hub 10 is $d_{10,12} = 4$. Since $d_{11,12} < d_{10,12}$, node 12 is selected for possible addition to the hub 11 pool, P_{11}. For (2), the smallest unit demand cost is $r_{14} = 2$ and hub 14 is selected for possible additional node. Since $d_{11,12} = r_{14}$, node 12 is connected to hub 11 and node 14 is selected to be a hub, as displayed in Figure 4.14. If these two are the same node, this same node is selected to be a hub.

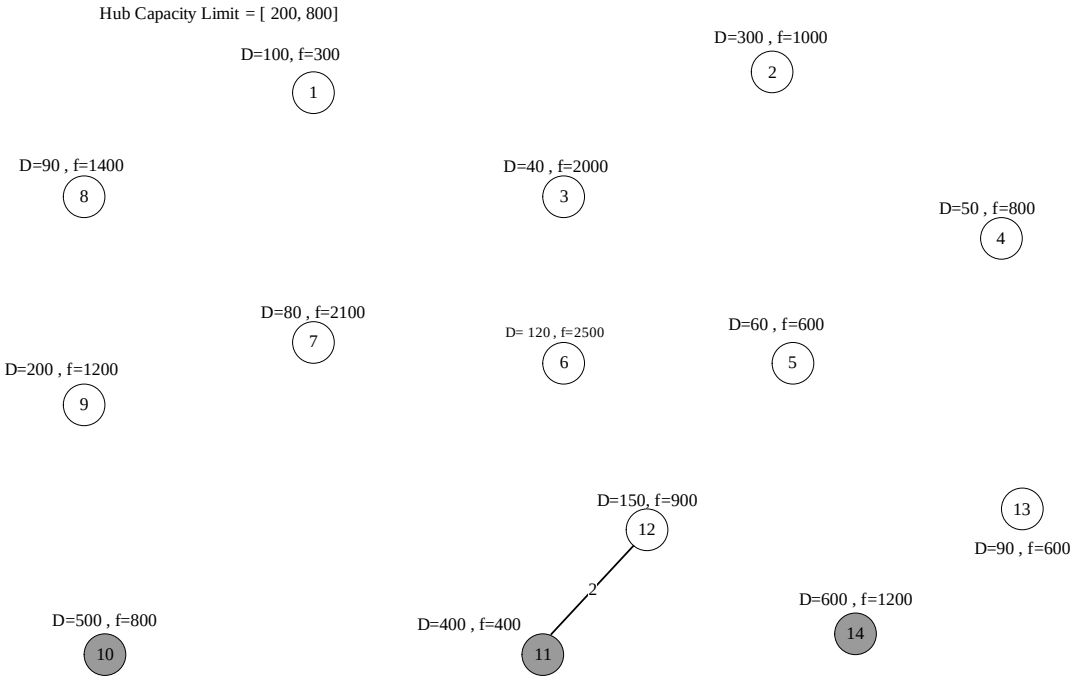

Figure 4.14. Node 12 is connected to hub 11 and node 14 is selected to be a hub

[IV]

For (1), the possible selection for the shortest distance for hub 11 is $d_{11,6} = 3$, hub 10 is $d_{10,9} = 5$, and hub 14 is $d_{14,13} = 3$. Since $d_{11,6} = d_{14,13} < d_{10,9}$, node 6 and node 13 are selected for possible additional nodes.

For (2), the smallest unit demand cost is $h(12; 6, 11)$ =2.716 and pool 12 is picked first. Pool region 12, including hub 12, node 6, and node 11, has the smallest value, 2.716. However, nodes 12 and 11 have been previously selected. The lower bound of pool 11 is not satisfied. Based on Step 9 (c), the total number of nodes at pool 12, TN_{12}, is 3 (hub 12, node 6, node 11), and at pool 11, TN_{11}, is 2 (hub11, node 12). Since $TN_{12} > TN_{11}$, $\frac{UDC_{j_1'}}{UDC_{j_2'}} \le \frac{\sum_k TD_{j_1'}^k}{\sum_k TD_{j_2'}^k}$ needs to be inspected. The value of $\frac{UDC_{12}}{UDC_{11}}$ is equal to $\frac{(f_{12} + D_6 d_{12,6} + D_{11} d_{12,11}) / (D_{12} + D_6 + D_{11})}{(f_{11} + D_{12} d_{11,12}) / (D_{11} + D_{12})} = \frac{2.716}{(400+150\times 2)/(400+150)} = 2.134$. The value of $\frac{\sum_k TD_{12}^k}{\sum_k TD_{11}^k}$ is $\frac{D_{12} + D_6 + D_{11}}{D_{11} + D_{12}} = \frac{150 + 120 + 400}{400 + 150} = 1.218$. Since 2.134 >1.218, $h(12; 6, 11)$ cannot be selected. Hence, the next candidate hub is sought. Pool region 1, which includes hub1, node 7, and node 3, and has the unit demand cost $h(1; 7, 3)$ =3, is found and selected for possible additional nodes.

Since $d_{11,6} = d_{14,13} = h(1; 7, 3)$ and all of them include different nodes, all of them are selected, as displayed in Figure 4.15.

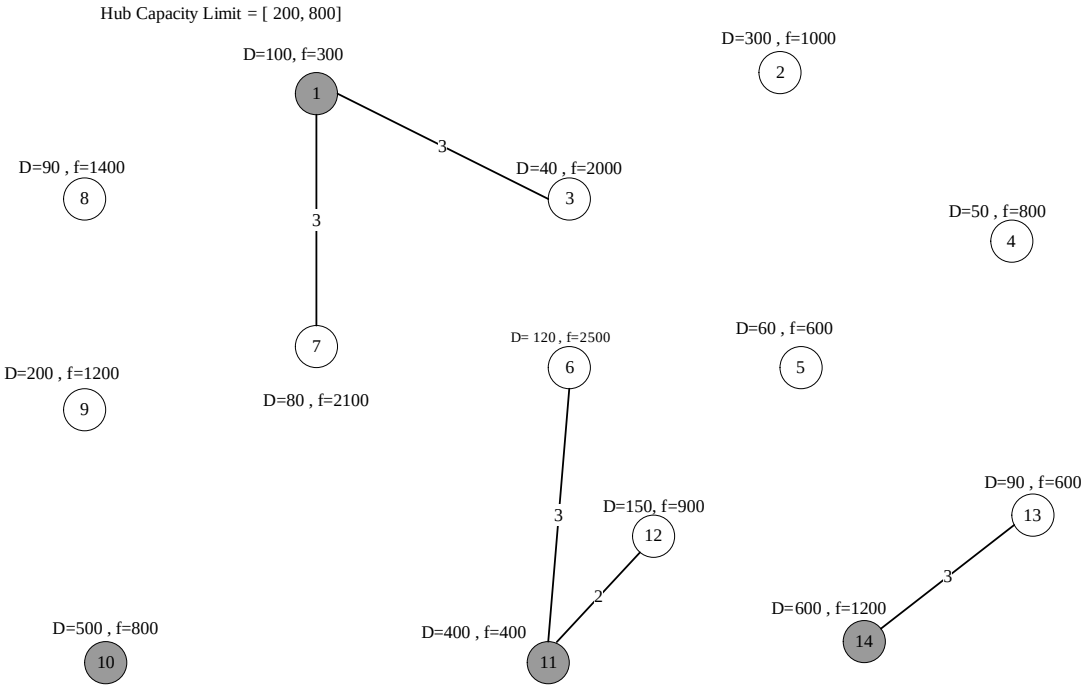

Figure 4.15. Node 6 is connected to hub 11, node 13 is connected to hub 14, and node 1 is selected to be a hub

[V]

For (1), the possible selection for the shortest distance for hub 11 is $d_{11,5} = 5$, hub 10 is $d_{10,9} = 5$, hub 14 is $d_{14,5} = 4$, and hub 1 is $d_{1,8} = 4$. Since $d_{14,5} = d_{1,8} < d_{11,5} = d_{10,9}$, node 5 and node 8 are selected for possible additional nodes.

For (2), the smallest unit demand cost is $h(2;3) = 3.176$ and hub 2 is picked first. Pool region 2, including hub 2 and node3, has the smallest value = 3.176. However, if $h(2;3)$ is selected, that will cause hub 1 not to satisfy the lower bound. Based on Step 9(c), the total number of nodes at pool 2, TN_2, is 2 (hub2, node 3), and at pool 1, TN_1, is 3 (hub1, node 3, node 7). Since $TN_2 < TN_1$, $h(2;3)$ cannot be selected. The next candidate pool, which only includes hub 2 and its unit demand cost r_2 is 3.33, is selected.

Since $r_2 < d_{14,5} = d_{1,8}$, node 2 is selected to be a hub, as shown in Figure 4.16.

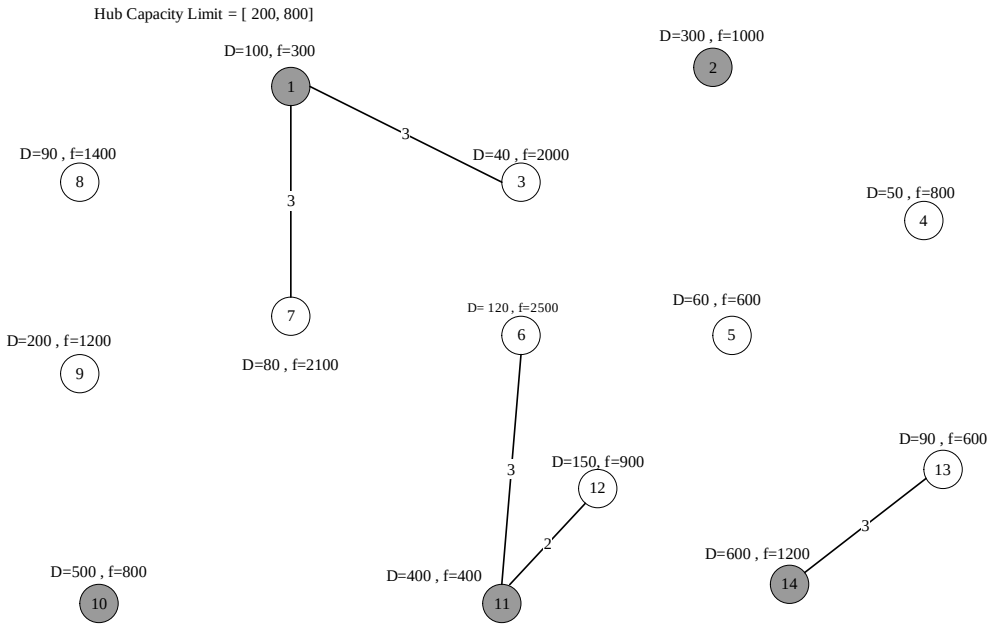

Figure 4.16. Node 2 is chosen to be a hub

[VI].

For (1), the possible selection for the shortest distance for hub 11 is $d_{11,5}=5$, hub 10 is $d_{10,9}=5$, hub 14 is $d_{14,5}=4$, hub 1 is $d_{1,8}=4$, and hub 2 is $d_{2,4}=4$. (Note: node 3 is not connected to hub 2 because that will cause hub 1 to violate its lower bound.) Since $d_{14,5}=d_{1,8}=d_{2,4}<d_{11,5}=d_{10,9}$, nodes 5, 8, and 4 are selected for possible additional nodes.

For (2), the smallest unit demand cost is $h(13;14)=3.478$ and hub 13 is picked first. Pool region 13, including hub 13 and node 14, has the smallest value=3.478. However, if $h(13;14)$ is selected, that will cause hub 14 not to satisfy the lower bound. Based on Step 9(c), the total number of nodes at pool 13, TN_{13}, is 2 (hub13, node 14), and at pool 14, TN_{14}, is 2 (hub 14, node 13). Since $TN_{13}\geq TN_{14}$, $\frac{UDC_{j_1'}}{UDC_{j_2'}}\leq\frac{\sum_k TD_{j_1'}^k}{\sum_k TD_{j_2'}^k}$ needs to be inspected. The value of $\frac{UDC_{13}}{UDC_{14}}$ is equal to $\frac{(f_{13}+D_{14}d_{13,14})/(D_{13}+D_{14})}{(f_{14}+D_{13}d_{14,13})/(D_{14}+D_{13})}=\frac{3.478}{(1200+90\times3)/(600+90)}=1.633$. The value of $\frac{\sum_k TD_{13}^k}{\sum_k TD_{14}^k}$ is $\frac{D_{13}+D_{14}}{D_{14}+D_{13}}=1$. Since 1.633 >1, $h(13;14)$ cannot be selected.

Hence, the next candidate hub is sought. Pool region 12, which includes hub 12 and node 6, and its unit demand cost $h(12;6)=3.778$. If $h(12;6)$ is selected, hub 11 still satisfies the lower bound. Based on Step 9 (b), $\frac{UDC_{12+11}}{UDC_{11}}-\frac{\sum_k TD_{12+11}^k}{\sum_k TD_{11}^k}=$

$$\frac{(f_{12}+f_{11}+D_6 d_{12,6})/(D_{12}+D_{11}+D_6)}{(f_{11}+D_6 d_{11,6}+D_{12}d_{11,12})/(D_{12}+D_{11}+D_6)}-\frac{(D_{12}+D_{11}+D_6)}{(D_{12}+D_{11}+D_6)}=\frac{3.778}{(400+120\times 3+150\times 2)/(400+120+150)}-1$$

$=1.582-1=0.582>0.$ $h(12;6)$ cannot be selected.

The next candidate pool region with the smallest unit demand cost is $h(5;6,12)=3.909$. Pool region 5, including hub 5, node 6, and node 12, has the smallest value=3.909. Based on Step 9(b), selecting $h(5;6,12)$ does not affect the lower bound for hub 11 but the value of

$$\frac{UDC_{5+11}}{UDC_{11}}-\frac{\sum_k TD^k_{5+11}}{\sum_k TD^k_{11}}=\frac{(f_{11}+f_5+D_6 d_{6,5}+D_{12}d_{12,5})\big/(D_{11}+D_5+D_6+D_{12})}{(f_{11}+D_6 d_{6,11}+D_{12}d_{12,11})\big/(D_{11}+D_6+D_{12})}-\frac{D_{11}+D_5+D_6+D_{12}}{D_{11}+D_6+D_{12}}$$

$$=\frac{(400+600+120\times 2+150\times 3)/(400+60+120+150)}{(400+120\times 3+150\times 2)/(400+120+150)}-\frac{(400+60+120+150)}{(400+120+150)}=1.463-1.090=0.373>0.$$

Hence, $h(5;6,12)$ cannot be selected.

The next available pool region 9 is found, which includes hub 9, node 7, and node 3 and its unit demand cost $h(9;7,3)$ =4.75. This value is greater than $d_{14,5}$, $d_{1,8}$, and $d_{2,4}$, which have the same value 4. Hence, node 5 is connected to hub 14, node 8 is connected to hub 1, and node 4 is connected to hub 2, as displayed in Figure 4.17.

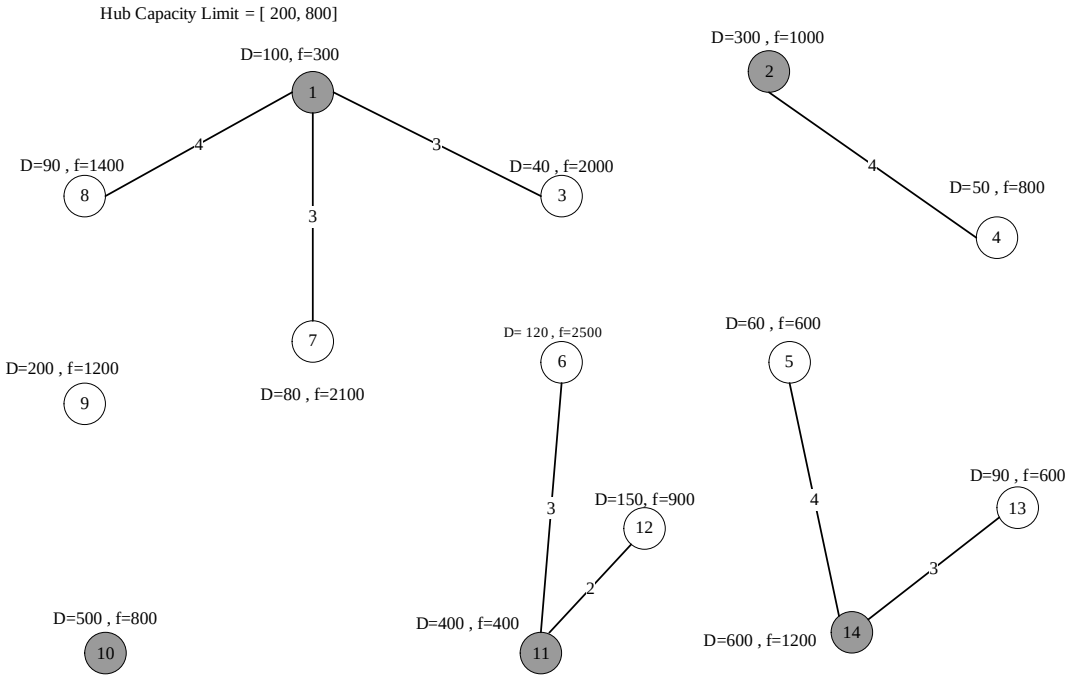

Figure 4.17. Node 5 is connected to hub 14, node 8 is connected to hub 1, and node 4 is connected to hub 2.

[VII]

For (1), the possible selection for the shortest distance for hub 11 is $d_{11,9} = 7$, hub 10 is $d_{10,9} = 5$, hub 14 is $d_{14,9} = 10$, hub 1 is $d_{1,9} = 5$, and hub 2 is $d_{2,3} = 2$. Since $d_{2,3}$ has the shortest distance, node 3 is selected for the possible additional node.

For (2), the smallest unit demand cost is $h(9; 7, 3) = 4.75$ and hub 9 is selected for the possible additional node.

Since $h(9; 7, 3) > d_{2,3}$, node 3 is connected to hub 2, as displayed in Figure 4.18.

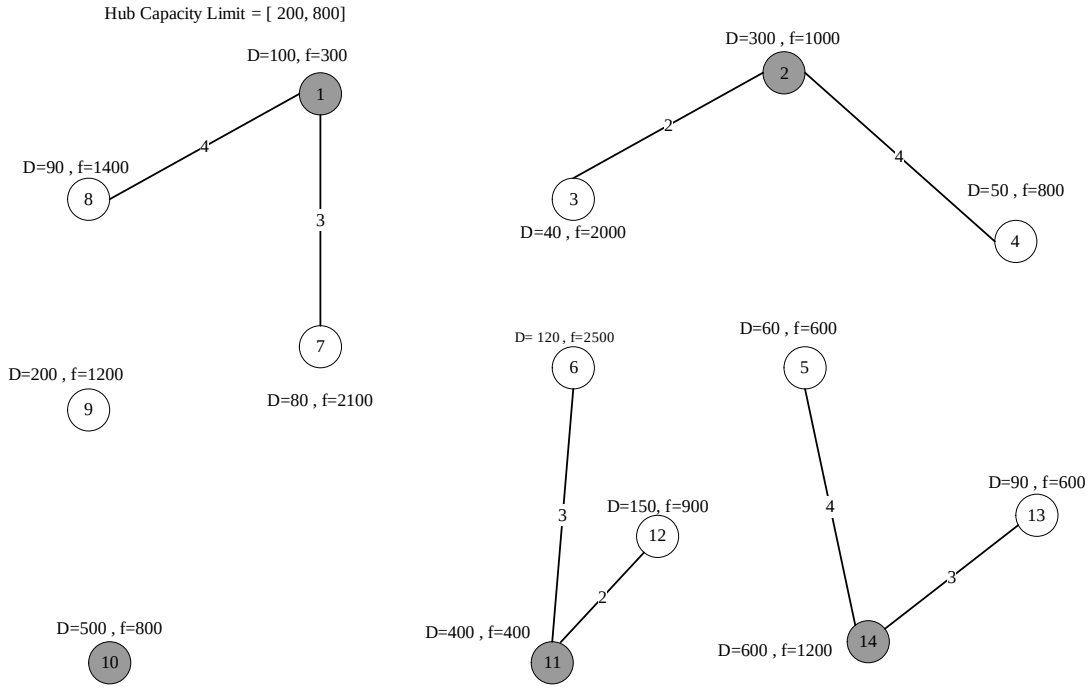

Figure 4.18. Node 3 is connected to hub 2

[VIII].

For (1), the possible selection for the shortest distance for hub 11 is $d_{11,9} = 7$, hub 10 is $d_{10,9} = 5$, hub 14 is $d_{14,9} = 10$, hub 1 is $d_{1,9} = 5$, and hub 2 is $d_{2,9} = 6$. Since $d_{10,9}$ and $d_{1,9}$ have the same shortest distance, they both are selected for possible additional nodes.

For (2), the smallest unit demand cost is $h(9; 7, 3) = 4.75$ and hub 9 is picked first. Pool region 9, including hub 9, node 7, and node 3, has the value of 4.75. However, if pool region 9 is chosen, that will result in hub 1 violating its lower bound but hub 3 still satisfies the lower bound. Based on Step 9(c), the total number of nodes at pool 9 and pool 2, TN_{9+2}, is 5 (hub 9, node 7, node 3, hub 2, node 4), and at pool 1 and pool 2, TN_{1+2}, is 6 (hub1, node 7, node 8, hub 2, node 3, node 4). Since $TN_{9+2} < TN_{1+2}$, $h(9; 7, 3)$ is not selected.

The next available pool region includes hub 9 and node 7 and its unit demand cost $h(9; 7)$ equals 4.857. However, it also will result in hub 1 violating its lower bound. Based on 9(c), the total number of nodes at pool 9, TN_9, is 2 (hub 9, node 7), and at pool 1, TN_1, is 3 (hub 1, node 7, node 8). Since $TN_9 < TN_1$, $h(9; 7)$ is also not selected.

The next available pool region includes hub 3, node 2, node 7, node 1, and node 9, and its unit demand cost is $h(3; 2, 7, 1, 9)$ equals 5.361. However, it also will result in hub 1 and hub 2 violating the lower bound. Based on 9(c), the total of nodes at pool 3, TN_3, is 5 (hub 3, node 2, node 7, node 1, node 9), and at pool 1 and pool 2, TN_{1+2}, is 6 (hub 1, node 7, node 8, hub 2, node 3, node 4). Since $TN_3 < TN_{1+2}$, $h(3; 2, 7, 1, 9)$ is not selected.

Since $h(3;2,7,1,9) > d_{10,9} = d_{1,9} = 5$, the next available pool region does not need to be checked. Either $d_{10,9}$ or $d_{1,9}$ is chosen. Hence, node 9 is connected to hub 10, as shown in Figure 4.19, or connected to hub 1, as shown in Figure 4.20. Either one can be selected for the next procedure. However, all nodes have been chosen, which means that either Figure 4.19 or Figure 4.20 is the solution. Both of them have the same smallest cost.

LINGO was utilized to verify the quality of this small example; the result shows that the solution from the clustering algorithm is the same as the optimal solution.

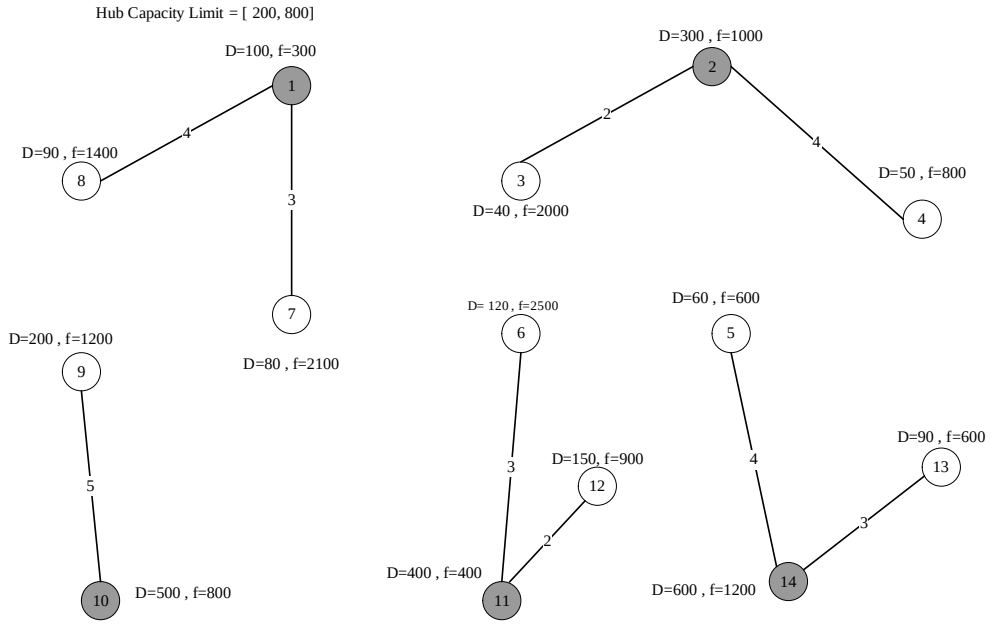

Figure 4.19. Node 9 is connected to hub 10

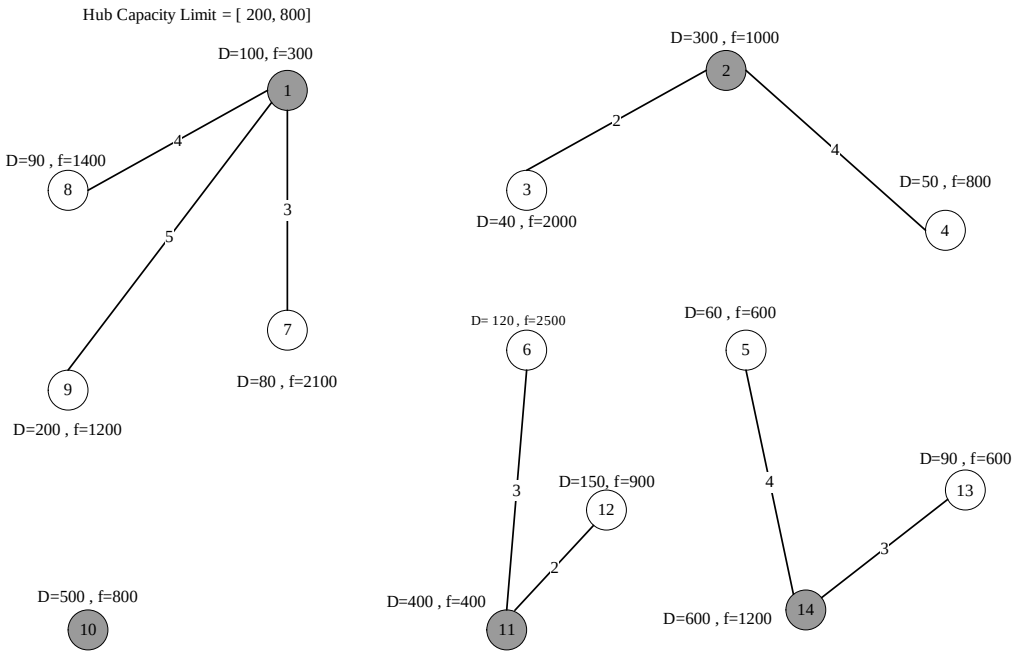

Figure 4.20. Node 9 is connected to hub 1

4.4.2 Enumeration Method

In this phase, the hubs *j* are unknown, but the pool regions are known. One hub is selected from each pool region. The total cost is calculated for each possible hub and then the hub with the lowest cost is chosen. The same rule is followed until all hubs are chosen.

4.4.3 Modified Prim's Algorithm

Step 1: For all nodes i excluding hub j, sort the total demand $\sum_{k=1}^{k_{\max}} D_i^k$ and rank these nodes i in descending order of $\sum_{k=1}^{k_{\max}} D_i^k$, denoted as DL.

Step 2: For each node i excluding hub j, sort the distance d_{ij} and rank these hubs j in ascending order of d_{ij}, denoted as dL_i.

Step 3: For these nodes which are not yet selected from DL, select the node i with the largest demand $\sum_{k=1}^{k_{\max}} D_i^k$ and assign it to the hub j with the shortest distance d_{ij}, selected from dL_i. The upper capacity needs to be satisfied simultaneously. Otherwise, select the next available hub from dL_i. Continue Step 3 until all nodes are assigned.

Step 4: Check if the lower capacity is satisfied for each pool region. If pool region j_1 (hub j_1) is not satisfied, for nodes i unassigned to pool region j_1, find the assigned pool region j_2 of node i and calculate the cost difference for node i, $diff_i = (d_{ij_1} - d_{ij_2}) \times \sum_{k=1}^{k_{\max}} D_i^k$. Sort the cost difference $diff_i$ and rank these nodes i in ascending order of $diff_i$, denoted as $diffL$. The upper capacity needs to be maintained simultaneously for pool region j_2. Otherwise, select the next available node from $diffL$. Continue Step 4 until the lower and upper capacities are satisfied for all pool regions.

4.5 Computing Results

This section focuses on an experimental design generated by four types of experimental factors including: the number of car types, the number of locations, pool capacity, and hub opening costs. Section 4.5.1 introduces the parameter settings and the meaning of the parameters. In Section 4.5.2, the setting of factor levels and three parts of the experiment are described. This description includes the problem size in practice and the problem size which can be solved optimally in a reasonable computing time by a branch-and-bound algorithm. The computer equipment used for conducting this experiment is introduced in Section 4.5.3. In Section 4.5.4, experimental results are analyzed through a three-part experiment.

4.5.1 Parameter Settings

The parameter settings in pool segmentation and hub selection are chosen based on the website data of Auto Rental News or the assumptions of making the problems feasible, and are as described below.

- ✧ **Distance**: The facility locations are randomly generated within a square space of 100×100. The Euclidean distance between two nodes is taken as the shortest distance in this experiment and is rounded off to five decimal places. Thus, the distances automatically satisfy the triangle inequality and are symmetric.
- ✧ **Demand**: The demand is a parameter in this experiment. Because the demand for small sized cars is normally higher than for the larger models at the same location, the demands for different sizes have the characteristic of this dependence. Hence, when a problem is generated based on a specified demand level, which equals 40, based on the

website data of Auto Rental News in 2007, the demand for car type 1 at each location is generated by multiplying this specified demand level by a random number generated from a uniform distribution (0.4, 5). As for the demands for car type 2 and the higher car types, a similar ratio value is generated from a uniform distribution (0.5, 1.2) for each car type. The demand for car type 2 is then generated by multiplying the demand of car type 1 by the ratio for car type 2. The demands for the higher car types are generated by multiplying the demand for the previous car type by the respective ratio. The demand is rounded off to an integer.

- **Lower Pool Capacity**: The lower pool capacity of car type 1 is set to be the specified demand level, 40. The lower capacity bound of other car types is generated by multiplying the demand for each car type by a random number generated from a uniform distribution (0.8, 1). This value is set to be low in order to make the feasible region larger. The lower pool capacity is rounded off to an integer.
- **Hub Opening Cost**: The ratio of hub opening cost to the demand is a factor in this experiment. Hence, when a problem is generated based on a specified demand level, 40, and a specified factor level of the ratio of hub opening cost to the demand, the hub opening cost is generated by multiplying the demand for car type 1 by a random number generated from a uniform distribution (0.5, 2) along with the ratio of hub opening cost to the demand. The hub opening cost is rounded off to an integer.
- **Neighbor Factor**: Because a value of neighbor factor m covers the neighbor indexes from 1 to m, a larger neighbor factor can extend the ranking list of all candidate pool regions. However, a larger neighbor factor also takes more computing time. Hence, it is important to strike a balance between the computing time and a larger ranking list. In

this experiment, the neighbor factor is set to be 5 in parts 1 and 3, but it is set to be 1 for the large cases in part 2.

- ✧ **Tabu List**: The Tabu List is set to be 200 based on the results after testing for several examples.

4.5.2 Factor Levels

Three parts of the experimental designs are utilized in pool segmentation and hub selection. In part 1, the large problem sizes are tested to determine which factors can significantly affect the algorithm time. In part 2, the number of locations in practical problem sizes is tested and compared to the algorithm time. In part 3, the problem size, which can be solved optimally in a reasonable time by a branch-and-bound algorithm, is tested to compare the solution quality and computing times of the clustering-based iterative algorithm and the optimal solution.

Based on the experimental design in part 1, the following 4 factors are tested to determine if the computing time is significantly affected.

- ✧ **Number of Locations:** Three factor levels are set to be 300, 600, and 1200. These numbers are close to the levels of the top 8, top 6, top 3 rental companies in the United States.
- ✧ **Number of Car Types:** Based on the website data of large car rental companies, three factor levels are set to be 4, 8, and 12.
- ✧ **Ratio of Pool Capacity to the Demand (R_Capacity):** The lower capacity bound is fixed to be the base demand for car type 1 at one location. The upper capacity bound is

set to be 10 times, 25 times, and 62.5 times the lower capacity. Three factor levels are set to be 10, 25, and 62.5.

- ✧ **Ratio of Hub Opening Cost to the Demand (R_Hub Cost):** Three factor levels are set to be 300, 600, and 1200.

In part 2, the number of locations in practical problem sizes is tested and compared to the algorithm time. The number of car types, the ratio of pool capacity to the demand, and the ratio of hub opening cost to the demand are fixed to be 8, 25 and, 600. The numbers of locations tested are 1000, 2000, 3000, 4000, 5000, and 6000. The number 6000 is close to the factor level of the number of locations of Enterprise Rent-A-Car, the top car rental company in the United States.

In part 3, the solution quality of the clustering-based iterative algorithm is compared to the optimal solution solved by a branch-and-bound algorithm. In addition, the computing time of the clustering-based iterative algorithm is compared. The problem size is assumed to be solved in a reasonable time by a branch-and-bound algorithm because the computing time of branch-and-bound is time-consuming. Three factors are tested and the number of car types is assumed to be 8.

- ✧ **Number of Locations:** Three factor levels are set to be 30, 60, and 90.
- ✧ **Ratio of Pool Capacity to the Demand:** The lower capacity bound is fixed to be the base demand for car type 1 at one location. The upper capacity bound is set to be 10 times, 15 times, and 20 times the lower capacity. Three factor levels are set to be 10, 15, and 20.
- ✧ **Ratio of Hub Opening Cost to the Demand:** Three factor levels are set to be 300, 600,

and 1200.

4.5.3 Experimental Platform

This experiment uses the software Visual C++ to compile the computer coding of the clustering-based iterative algorithm and uses the optimization software LINGO 9.0 to find an optimal solution. The computer equipment utilized to conduct this experiment includes an Intel Core 2 Duo E7400 2.80 GHz CPU and 6 GB memory.

4.5.4 Experimental Analysis

The experiment is divided into three parts. Four types of experimental factors are utilized to test which factors affect the computing time in part 1. Each factor contains three levels and each factor level uses three replications based on different random seeds. Hence, 3 factor levels of the number of locations $\times 3$ factor levels of the number of car types $\times$ 3 factor levels of the ratio of the capacity to the demand $\times$ 3 factor levels of the ratio of hub opening cost to the demand $\times$ 3 random seeds $=$ 243 independent trials in part 1.

In part 2, the number of locations is the only experimental factor and it contains six factor levels. Hence, 6 factor levels of the number of locations $\times$ 3 random seeds = 18 independent trials in part 2.

In part 3, an experiment is conducted with the problem size that was solved in a reasonable time by the branch-and-bound algorithm. The clustering-based iterative algorithm is compared to the optimal solution solved by the branch-and-bound algorithm. Three types of experimental factors are utilized to test which factors affect the solution quality of the

algorithm. Each factor contains three levels and each factor level conducts three replications based on different random seeds. Hence, 3 factor levels of the number of locations × 3 factor levels of the ratio of the capacity to the demand × 3 factor levels of the ratio of hub opening cost to the demand × 3 random seeds = 81 independent trials in part 3.

The Statistics Software Minitab is used to implement a General Linear Model to run the Analysis of Variance (ANOVA) in order to analyze the experimental factors versus the algorithm time and the solution gap. The significance level is set to be 5%. If the main effect of a specific experimental factor is significant, this means that the algorithm time has a significant difference among different levels of this specific experimental factor. Hence, the Tukey's test is conducted to find which means are significantly different from one another in this specific experimental factor.

4.5.4.1 Impact Analysis on Experimental Factors versus Algorithm Time

In the results of part 1, the statistical distribution of the residuals of the algorithm time is not a normal distribution. Data transformation of the logarithm of the algorithm time is applied to resemble a normal distribution. Hence, the logarithm of the algorithm time is used as the response. The original results of this part can be referenced in Appendix A.

Observe in Table 4.4 that all *p*-values of these four factors are less than 0.05. That means that the effects of these four factors are all significant. Hence, the number of car types, the number of locations, the ratio of pool capacity to the demand, and the ratio of hub cost to the demand can affect the computing time of the algorithm.

Table 4.4. ANOVA table on four factors versus algorithm time

Source	DF	SSE	MSE	F value	p value
Car Types	2	0.777	0.389	20.22	0.0
Locations	2	136.097	68.049	3538.75	0.0
R_Capacity	2	3.157	1.578	82.09	0.0
R_HubCost	2	3.926	1.963	102.08	0.0
Error	234	4.500	0.019		
Total	242	148.457			

Next, Tukey's test is conducted to compare different factor levels on each experimental factor. The original results of Tukey's test are represented in Appendix B. The summary data are shown from Table 4.5 to Table 4.8. The levels of experimental factors are represented in Column 1. Column 2 is the number of experimental trials. Column 3 shows whether or not the responses on different levels of the factors are significantly different. If the responses on different levels of the factors are divided into different groups, this means that their responses are significantly different. The group type of the algorithm time is named in alphabetical order. The closer to *A* the letter is in the alphabet, the faster the algorithm time, and that is $A > B > C > \ldots$. If two factor levels are divided into the same group, this means that there is not a significant difference between the algorithm times of these two factor levels. The value of the group *A, B, C* is the average logarithm algorithm time in the level of experimental factor. The smaller the value, the faster the algorithm time.

Tukey's test on different numbers of car types is represented in Table 4.5. In Table 4.5, there are significant differences in the algorithm time between the numbers of car types, 4 and 8, or, 4 and 12. The algorithm time of car type 4 is slower than the algorithm time of car type 8 or 12. However, there seems not to be any difference in the algorithm time between the numbers of car type 8 and 12.

Table 4.5. Tukey test on different numbers of car types

Car Type	# of Trials	Group (log time)	
		A	B
4	81		1.3246
8	81	1.2110	
12	81	1.1991	

Tukey's test on different numbers of locations is represented in Table 4.6. From Table 4.6, the greater the number of locations, the slower the algorithm time.

Table 4.6. Tukey test on different numbers of locations

Locations	# of Trials	Group (log time)		
		A	B	C
300	81	0.3318		
600	81		1.2381	
1200	81			2.1649

Tukey's test on different ratios of pool capacity to the demand is represented in Table 4.7. From Table 4.7, the tight pool capacity seems to make the algorithm time much longer. In addition, the loose pool capacity results in a longer algorithm time.

Table 4.7. Tukey test on different ratios of pool capacity to the demand

R_Capacity	# of Trials	Group (log time)		
		A	B	C
10	81			1.3646
25	81	1.0915		
62.5	81		1.2786	

Tukey's test on different ratios of hub cost to the demand is represented in Table 4.8. From Table 4.8, the larger the ratio of pool capacity to the demand, the slower the algorithm time.

Table 4.9. Tukey test on different ratios of hub cost to the demand

R_HubCost	# of Trials	Group (log time)		
		A	B	C
300	81	1.0910		
600	81		1.2415	
1200	81			1.4023

The figures of average logarithm time on different levels of these four experimental factors are summarized as in Figure 4.21

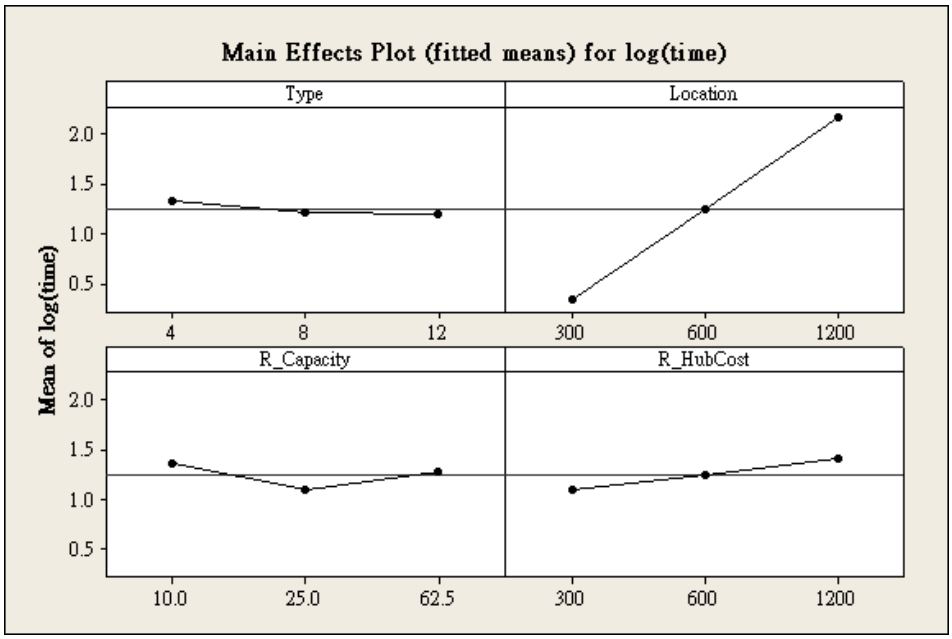

Figure 4.21. Average logarithm time on different levels of experimental factors

4.5.4.2 Impact Analysis on Locations in Practical Problem Size versus Algorithm Time

The experiment of part 2 is the impact analysis on the number of locations in practical problem size versus algorithm time. The number of locations is the most important experimental factor in this problem because it affects how large of a problem size this algorithm can solve. Table 4.9 represents the algorithm time and the number of integer variables on different numbers of locations and its trend chart is presented in Figure 4.22. The largest case, 6000, is similar to the number of locations in the top rental company in the United States, Enterprise Rent-A-Car. Thirty-six million integer variables are covered in this case and the algorithm takes about 135 minutes to solve.

Table 4.9. The algorithm time on the number of locations of practical problem size

Example	# of locations	# of integer variables (millions)	Seed #	Time (sec)	Avg time (sec)
1	1,000	1	20	23	20
2			40	18	
3			60	19	
4	2,000	4	20	171	169
5			40	200	
6			60	138	
7	3,000	9	20	800	645
8			40	590	
9			60	545	
10	4,000	16	20	2,183	1,564
11			40	1,297	
12			60	1,211	
13	5,000	25	20	4,490	3,340
14			40	2,943	
15			60	2,588	
16	6,000	36	20	10,301	8,112
17			40	7,111	
18			60	6,924	

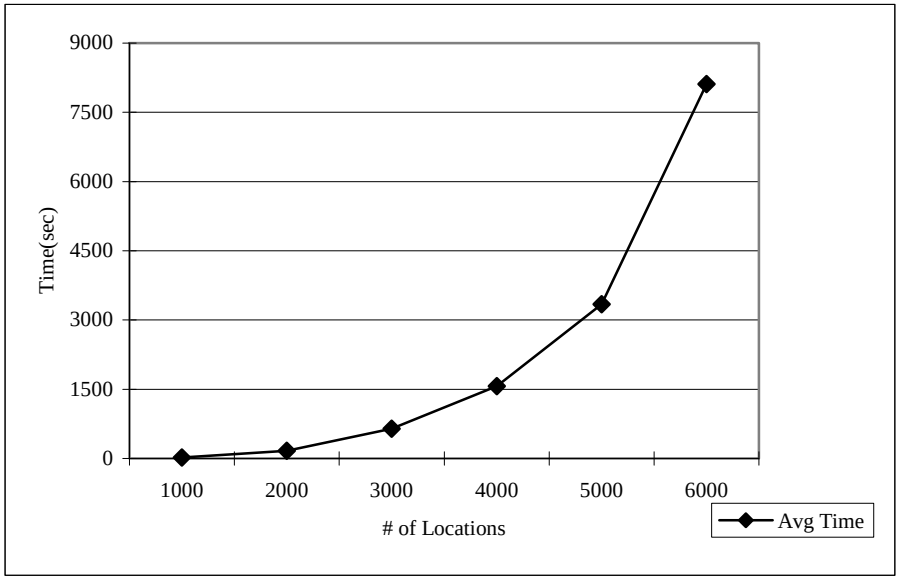

Figure 4.22. The trend chart of algorithm time on the number of locations of practical problem size

4.5.4.3 Impact Analysis on Experimental Factors versus Solution Gap

In the ANOVA table of part 3 shown in Table 4.10, all p-values of these three factors are less than 0.05. The effects of these four factors are all significant. Hence, the number of locations, the ratio of pool capacity to the demand, and the ratio of hub cost to the demand, can affect the solution gap. The original results of part 3 can be referenced in Appendix C.

Table 4.10. ANOVA table on three factors versus solution gap

Source	DF	SSE	MSE	F value	p value
Locations	2	0.0052972	0.0026486	8.78	0.000
R_Capacity	2	0.0061696	0.0030848	10.23	0.000
R_HubCost	2	0.0049541	0.0024771	8.22	0.001
Error	74	0.0223130	0.0003015		
Total	80	0.0387339			

Tukey's test is conducted to compare different levels of each experimental factor. The original results of Tukey's test are represented in Appendix D. The illustration of this process can be reference in Section 4.5.4.1.

Tukey's test on different numbers of locations is represented in Table 4.11. In Table 4.11, there are significant differences in the solution gaps between the numbers of locations, 30 and 60, or, 30 and 90. The solution gap of the number of locations, 30, is smaller than the solution gap of the number of locations 60 or 90. However, there seems not to be any significant difference in the solution gap between the numbers of locations 60 and 90.

Table 4.11. Tukey test on different numbers of locations

Locations	# of Trials	Group (Solution Gap)	
		A	B
30	27	1.10%	
60	27		2.46%
90	27		3.03%

Tukey's test on different ratios of pool capacity to the demand is represented in Table 4.12. From Table 4.12, there are significant differences in the solution gap between the ratios, 10 and 15, or, between the ratio, 10 and 20. The solution gap of the ratio of pool capacity to the demand, 10, is larger than the solution gap of the ratio of pool capacity to the demand, 15 or 20. However, there seems not to be any significant difference in the solution gaps between the ratio 15 and 20.

Table 4.12. Tukey test on different ratios of pool capacity to the demand

R_Capacity	# of Trials	Group (Solution Gap)	
		A	B
10	27		3.41%
15	27	1.80%	
20	27	1.39%	

Tukey's test on different ratios of hub cost to the demand is represented in Table 4.13. From Table 4.13, there is a significant difference in the solution gap between the ratios, 300 and 1200. The solution gap of the ratio of hub cost to the demand, 300, is smaller than the solution gap of the ratio of hub cost to the demand, 1200. However, there seems not to be any significant difference in the solution gap between the ratios, 300 and 600, or, 600 and 900.

Table 4.13. Tukey test on different ratios of hub cost to the demand

R_HubCost	# of Trials	Group (Solution Gap)		
		A	B	C
300	27	1.28%		
600	27	2.13%	2.13%	
1200	27		3.19%	3.19%

The figures of average solution gap on different levels of these three experimental factors are summarized as in Figure 4.23.

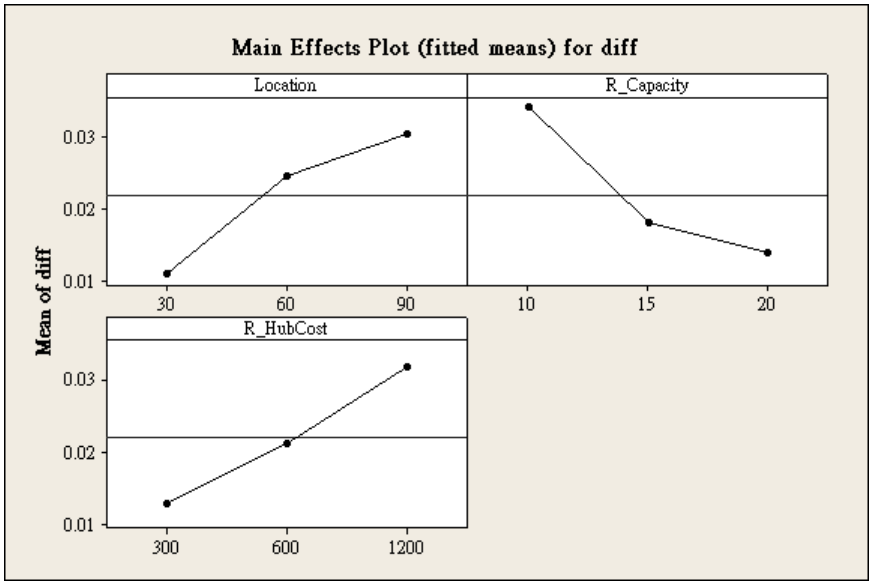

Figure 4.23. Average solution gap on different levels of experimental factors

4.5.4.4 Comparison of Computing Time Between the Clustering-Based Iterative Algorithm and the Branch-and-Bound Method

In order to more easily measure the computing time of different problems, the computing time and the objective value is standardized. The computing time t^* of the best solution found by the branch-and-bound method in LINGO Software in each problem is set to be $\log_{10}(10^6 * \frac{t^*}{t^*}) = 6$. Other computing times t can be transformed to standardized logarithm time $\log_{10}(10^6 * \frac{t}{t^*})$. This time setting can avoid the value of standardized logarithm time to be negative and the scale of logarithm time is easily compared to multiple solutions, especially shown in the same plot. In this problem, Algorithm Solution, BetterSolu, and BestSolu are included.

If the optimal objective value found by the branch-and-bound method is Z^* and the objective value of the clustering-based iterative algorithm is Z, the solution gap is calculated by $\frac{Z - Z^*}{Z^*} \times 100\%$. These standardized data are used in the following observations and analysis.

The comparison of the computing time between the clustering-based iterative algorithm and the branch-and-bound method is presented in Table 4.14. In Table 4.14, three kinds of computing times are recorded. Algorithm times in Column 5 and Column 8 represent average times and average logarithm times of the clustering-based iterative algorithm, respectively. BetterSolu time is recorded when a better solution than in the clustering-based iterative algorithm is found in LINGO. BetterSolu times in Column 6 and Column 9 represent average times and average logarithm times when a better solution found. BestSolu time t^* represents the time of the optimal solution found in LINGO.

If the gap of two logarithm times is 1, this means that the gap of these two computing times is 10 times. In Table 4.14, the overall results show that the average gaps of the logarithm times between the algorithm time and better solution fall between 2 and 3 and the average gaps of the logarithm times between the algorithm time and best solution fall between 2 and 4. This means that the branch-and-bound method usually takes 100~1,000 times the algorithm time to find a better solution and 100~10,000 times the algorithm time to find an optimal solution than in the clustering-based iterative algorithm.

Table 4.14. Computing time of the clustering-based iterative algorithm and the branch-and-bound method

Locations	R_Cap	Ratio Hubcost	# of trials	Avg Time (sec)			Avg Log Time		Solution Gap (%)
				Algorithm Time	BetterSolu Time	BestSolu Time	Algorithm Time	BetterSolu Time	
30	10	300	3	0.065	2.667	2.667	4.03	6.00	0.46%
		600	3	0.025	5.667	7.000	3.39	5.93	3.50%
		1200	3	0.075	5.333	8.667	3.63	5.85	2.27%
	15	300	3	0.011	3.000	3.000	3.52	6.00	0.16%
		600	3	0.012	2.333	2.333	3.71	6.00	0.52%
		1200	3	0.025	4.333	4.667	3.56	5.97	1.08%
	20	300	3	0.011	2.667	2.667	3.56	6.00	0.16%
		600	3	0.012	2.667	2.667	3.63	6.00	1.08%
		1200	3	0.008	2.333	2.333	3.51	6.00	0.72%
60	10	300	3	0.095	10.000	10.667	3.84	5.98	1.14%
		600	3	0.047	18.667	22.667	3.30	5.92	1.99%
		1200	3	0.042	24.667	261.667	2.40	5.20	5.79%
	15	300	3	0.027	7.667	7.667	3.56	6.00	0.64%
		600	3	0.027	19.000	24.667	3.07	5.91	2.39%
		1200	3	0.022	20.000	338.333	2.49	5.43	4.17%
	20	300	3	0.029	7.667	7.667	3.58	6.00	1.33%
		600	3	0.023	8.000	8.000	3.50	6.00	2.82%
		1200	3	0.025	14.000	28.667	3.10	5.80	1.90%
90	10	300	3	0.247	59.667	512.333	2.78	5.19	4.44%
		600	3	0.271	82.333	5309.667	2.05	4.58	3.78%
		1200	3	0.254	108.667	80826.333	0.53	3.17	7.32%
	15	300	3	0.144	33.333	36.333	3.53	5.96	1.35%
		600	3	0.099	51.000	142.000	2.93	5.64	2.10%
		1200	3	0.048	69.000	3230.333	1.98	5.09	3.80%
	20	300	3	0.074	13.333	13.333	3.73	6.00	1.85%
		600	3	0.044	20.000	22.000	3.36	5.98	0.96%
		1200	3	0.047	35.000	58.667	2.92	5.79	1.66%

The comparisons of logarithm time between the clustering-based iterative algorithm and the branch-and-bound method are represented from Figure 4.24 to Figure 4.26. Figure 4.24 is based on different numbers of locations. Figure 4.25 is based on different ratios of pool capacity to the demand. Figure 4.26 is based on different ratios of hub cost to the demand. From these three figures, the computing time of the clustering-based iterative algorithm seems always 100 times faster than the computing time of BetterSolu time.

In Figure 4.24, the greater the number of locations, the smaller the logarithm time of the clustering-based iterative algorithm and the BetterSolu. As a result of the proportional scale characteristic of the logarithm time $\log_{10}(10^6 * \frac{t}{t^*})$ compared to BestSolu time t^*, the logarithm time becomes less as the number of locations increases. This means that the efficiency of the clustering-based iterative algorithm is much better than the branch-and-bound method as the number of locations increases.

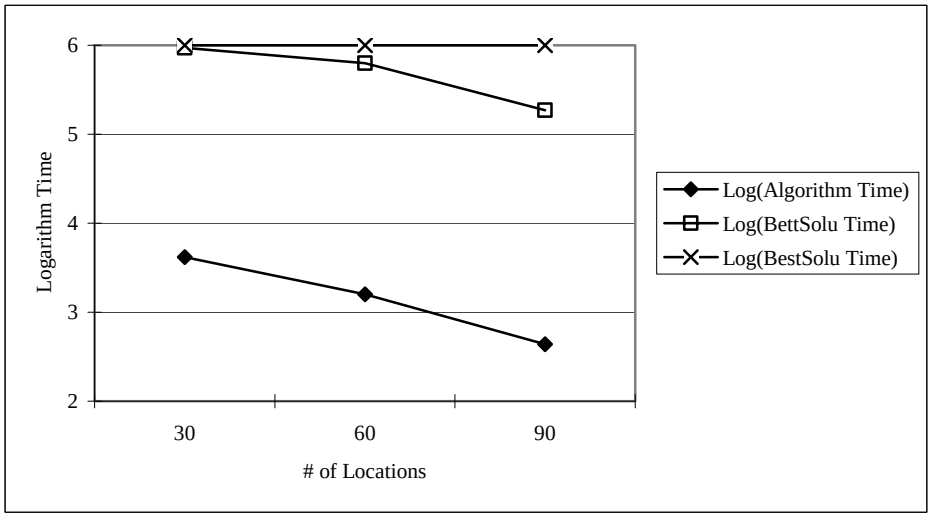

Figure 4.24 Comparison of logarithm time between the clustering-based iterative algorithm and the branch-and-bound method on different numbers of locations

In Figure 4.25, the larger the ratio of pool capacity to the demand, the larger the logarithm time of the clustering-based iterative algorithm and the BetterSolu. The means that the efficiency of the clustering-based iterative algorithm is much better than the branch-and-bound method as the ratio of pool capacity versus demand decreases.

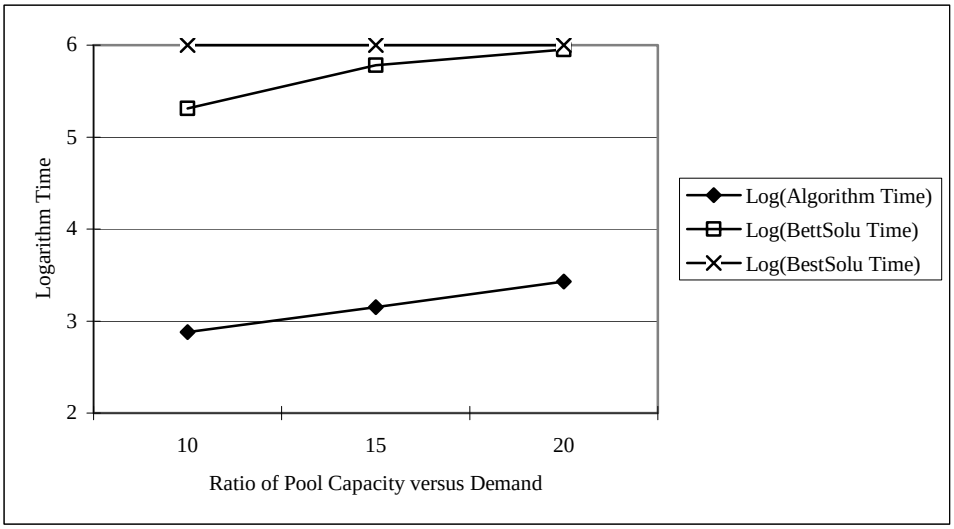

Figure 4.25 Comparison of logarithm time between the clustering-based iterative algorithm and the branch-and-bound method on different ratio of pool capacity to the demand

In Figure 4.26, the larger the ratio of hub cost to the demand, the smaller the logarithm time of the clustering-based iterative algorithm and the BetterSolu This means that the efficiency of the clustering-based iterative algorithm is much better than the branch-and-bound method as the ratio of hub cost versus demand increases.

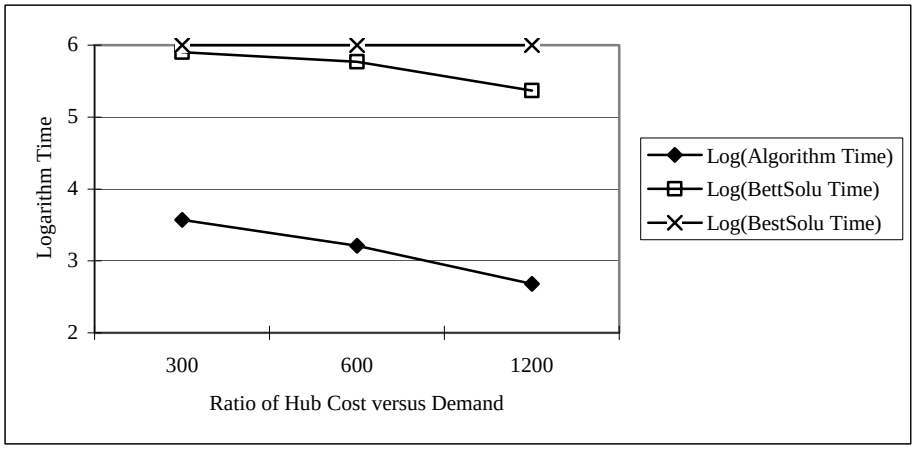

Figure 4.26 Comparison of logarithm time between the clustering-based iterative algorithm and the branch-and-bound method on different ratios of hub cost to the demand

4.6 Concluding Remarks

In this chapter, a model of pool segmentation and hub selection was introduced and a clustering-based iterative algorithm was proposed and validated. This algorithm utilizes three important modules. The clustering algorithm uses the concept of unit demand cost to cluster nearby locations and quickly captures a very good initial solution. The iterative procedure of an enumeration method and a modified Prim's algorithm utilizes the concept of a convex function to obtain a near-optimal solution.

Based on the numerical results, the computing time of the clustering-based iterative algorithm is sensitive to all experimental factors. Fewer car types, more locations, tight or loose ratios of pool capacity to the demand, and larger ratios of hub cost to the demand will lead to longer computing time. The algorithm times on different numbers of locations in practical problem sizes were also compared. The largest case was tested for 6000 nodes and 36 million integer variables, which is close to the level of Enterprise Rent-A-Car, the top car rental company in the United States. This algorithm takes about 135 minutes to solve.

Compared to the optimal solution solved by the branch-and-bound method, the branch-and-bound method needs 100~1,000 times the algorithm time to find a better solution and 100 ~10,000 times to find an optimal solution than in the clustering-based iterative algorithm. In addition, the solution gap of the clustering-based iterative algorithm is relatively small with an average gap of 2.22%. The numerical results show that the clustering-based iterative algorithm achieves a near-optimal solution in an extremely short time.

CHAPTER 5

INTER-POOL MOVES AND ASSET REPLACEMENT

5.1 Problem Formulation

In Chapter 4, all locations are allocated to different pools and one hub is selected for each pool based on the yearly demand, the distance cost, and the hub opening cost. When the job of pool segmentation and hub selection is accomplished for long-term planning, the next task is to distribute the inter-pool moves and asset replacement, which is selling and buying cars among different pool regions based on the change of seasonal demand. Seasonal demands, selling prices, buying prices, inventory costs, and transportation costs are assumed to be known. The inter-pool moves, and buying/selling cars are allocated to different pool regions as shown in Figure 5.1. In addition, the service level concerning the fraction of demand satisfied for different car types needs to be achieved.

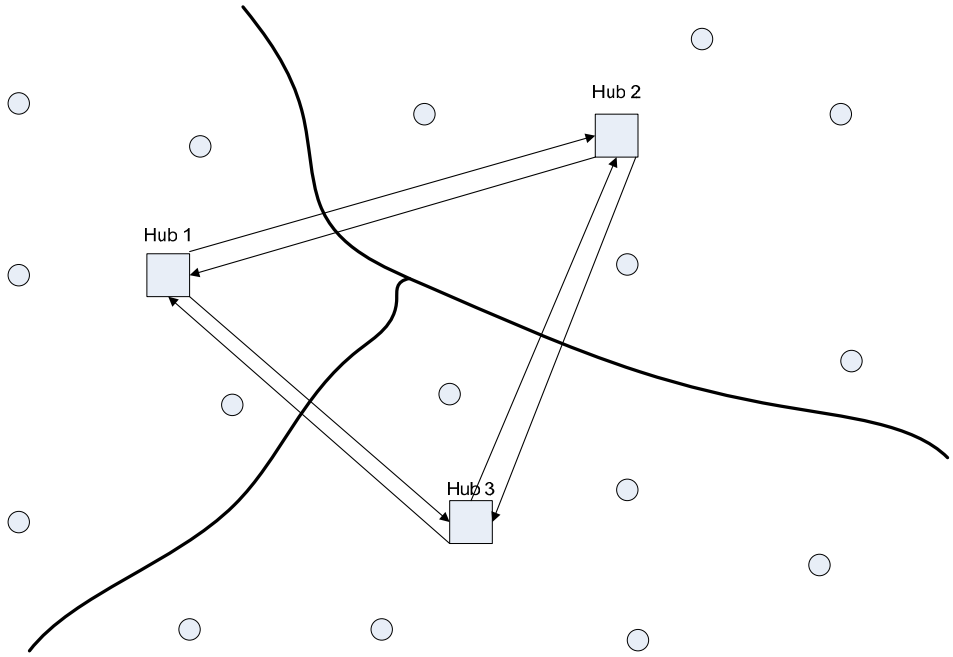

Figure 5.1. Network for inter-pool moves and asset replacement

5.2 Mathematical Model

In this section, the framework of a mathematical model for seasonal inter-pool moves, asset replacement (buying/selling cars), service level, and upgrade policy is proposed. This problem is formulated as an integer programming model.

Indices

l = pool l
t = seasonal period t
k = car type k
a = seasonal age a

Parameters

$d_{l,t}^{k}$ = seasonal forecasting demand for cars with a car type k in pool l at seasonal period t
$s_{l,t}^{k,a}$ = unit price for a car with a car type k and seasonal age a sold in pool l at seasonal period t
$tr_{l,l',t}^{k,a}$ = unit transportation cost for a car with car type k, seasonal age a from pool l to pool l' at seasonal period t
α^{k} = fraction of demand satisfied for a car type k
$b_{l,t}^{k}$ = unit price for a car with a car type k purchased in pool l at seasonal period t
$inv_{l,t}^{k,a}$ = unit cost for a car with a car type k, age a stored in pool l at seasonal period t

Variables

$I_{l,t}^{k,a}$ = number of cars with car type k and seasonal age a stored in pool l at seasonal period t

$B_{l,t}^{k}$ = number of cars with car type k purchased in pool l at seasonal period t

$S_{l,t}^{k,a}$ = number of cars with car type k and seasonal age a sold in pool l at seasonal period t

$X_{l,l',t}^{k,a}$ = number of cars with car type k and seasonal age a redistributed from pool l to pool l' at seasonal period t

The model can be expressed as:

$$Min \sum_{t}\sum_{l}\sum_{k}\sum_{a}\left[inv_{l,t}^{k,a} I_{l,t}^{k,a} - s_{l,t}^{k,a} S_{l,t}^{k,a} \right] + \sum_{t}\sum_{k}\sum_{l} b_{l,t}^{k} B_{l,t}^{k} + \sum_{t}\sum_{k}\sum_{a}\sum_{l}\sum_{l',l'\neq l} tr_{l,l',t}^{k,a} X_{l,l',t}^{k,a}$$

subject to:

$$\sum_{k=k'}^{k_{\max}}\sum_{a=0}^{a_{\max}} I_{l,(t-1)}^{k,a} + \sum_{k=k'}^{k_{\max}} B_{l,t}^{k} - \sum_{k=k'}^{k_{\max}}\sum_{a=1}^{a_{\max}} S_{l,t}^{k,a} + \sum_{k=k'}^{k_{\max}}\sum_{a=1}^{a_{\max}}\sum_{l',l'\neq l}^{l_{\max}} (X_{l',l,t}^{k,a} - X_{l,l',t}^{k,a})$$

$$\geq \alpha^{k'} \times \sum_{k=k'}^{k_{\max}} d_{l,t}^{k} \quad \forall l,t,k',k' = \{1,2,...,k_{\max}\} \qquad (5.2)$$

$$B_{l,t}^{k} - S_{l,t}^{k,a} + \sum_{l',l'\neq l} X_{l',l,t}^{k,a} - \sum_{l',l'\neq l} X_{l,l',t}^{k,a} = I_{l,t}^{k,a} \qquad \text{for a} = 1, \forall l,k,t \qquad (5.3)$$

$$I_{l,(t-1)}^{k,(a-1)} - S_{l,t}^{k,a} + \sum_{l',l'\neq l} X_{l',l,t}^{k,a} - \sum_{l',l'\neq l} X_{l,l',t}^{k,a} = I_{l,t}^{k,a} \qquad \text{for a} \neq 1, \forall l,k,t \qquad (5.4)$$

$I_{l,t}^{k,a}, B_{l,t}^{k}, S_{l,t}^{k,a}, X_{l,l',t}^{k,a}$ are integers $\qquad \forall l,l',k,a,t \qquad (5.5)$

$I_{l,t=0}^{k,a}$ is given $\qquad \forall l,k,a \qquad (5.6)$

Objective (5.1) is to minimize the total cost of inter-pool moves plus asset replacement. Constraints (5.2) indicate that at least the fraction $\alpha^{k'}$ of demand for car type k' and higher car types is satisfied. Constraints (5.3) and (5.4) are the inventory balance constraints. Constraints (5.5) are the integrality requirements and initial inventory levels are given in Constraints (5.6).

5.3 Motivation

In the design of the algorithm procedures in a very large scale integer programming problem, two kinds of algorithms should be considered. One type of algorithmic procedure, such as Benders' decomposition or Lagrangian relaxation, decomposes the complicating variables or constraints to obtain a solution bound or a LP solution. Due to the simplification of the original problem, the problem is easier to solve; however, if the original problem after decomposing is still a NP-hard or NP-complete problem, it will need other algorithmic procedures or meta-heuristics to solve the problem successfully. Another class of algorithms used for large scale problems is meta-heuristics. Due to the large number of integer variables, a possible direction is to exploit meta-heuristics to solve this combinatorial optimization problem. Some traditional meta-heuristics, such as Tabu Search (Glover 1986), Simulated Annealing (Kirkpatrick *et al.* 1983), Genetic Algorithms (Holland 1975), Memetic Algorithms (Moscato 1989), and Ant Colony Optimization (Dorigo *et al.* 1996) are widely used. In recent years, other search mechanisms, such as Scatter Search (Glover 1998), Variable Neighborhood Search (Mladenovic and Hansen 1997), Guided Local Search (Voudouris and Tsang 1996), Greedy Randomized Adaptive Search Procedure (Feo and Resende 1995), Iterated Local Search (Lourenc *et al.* 2002), and Nested Partition Method (Shi and Olafsson 2000) have also been developed. Most of the meta-heuristics are implemented in binary integer programming problems and their neighborhood structures are normally designed for binary variables by different moves, such as swap moves and insert moves. However, all variables in this study are non-binary integer variables, and are not suitable for swap or insert moves.

Therefore, to solve this problem in this study, a new neighborhood structure $Var \pm h$ called “better neighbors” is proposed. The better neighbors are obtained from the value of a specific variable *Var* adding/subtracting a flexible value *h,* and are only adopted for those which have better objectives than the current solution. A flexible value *h* is decided based on the maximal reduction of the objective. If a fixed value *h* is adopted for the design of a neighborhood structure, it will only reduce a fixed objective value in each iteration and need more iterations in the same type of neighbor. However, a flexible value *h* will reduce the number of iterations in the same type of neighbor because *h* is decided by the maximal reduction of the objective.

Among the four kinds of variables, $X_{l,l',t}^{k,a}$ includes five parameter indices, $S_{l,t}^{k,a}$ and $I_{l,t}^{k,a}$ have four, and $B_{l,t}^{k}$ has three. If one large problem covers 8 car types, 12 car ages, 12 seasonal periods, and 200 pool regions, there will be 8×12×200×(200-1)×12=45,849,600 integers variables in $X_{l,l',t}^{k,a}$, 8×12×200×12=230,400 integer variables in $S_{l,t}^{k,a}$ and $I_{l,t}^{k,a}$ respectively, and 8×200×12=19,200 integer variables in $B_{l,t}^{k}$. Of all integer variables, $X_{l,l',t}^{k,a}$ utilize 98.96%, $S_{l,t}^{k,a}$ and $I_{l,t}^{k,a}$ utilize 0.50% each, and $B_{l,t}^{k}$ only utilize 0.04%. However, $B_{l,t}^{k}$ has the largest impact on the objective. Next is $S_{l,t}^{k,a}$, and $I_{l,t}^{k,a}$ and $X_{l,l',t}^{k,a}$ have the smallest impact on the objective. Their impacts on the objective are in inverse proportion to the numbers of integer variables.

Since the change of any integer variables will force other integer variables to change because of the inventory balance constraints, the design of this algorithm will be based on the change of a single integer variable. All neighbors of $B_{l,t}^{k}, S_{l,t}^{k,a}, X_{l,l',t}^{k,a}$ are not calculated and

evaluated at the same iteration because that will take an enormous amount of time to compute. Instead, the neighbors of the variables $B_{l,t}^{k}$, which have the largest impact on the objective, are first evaluated. $B_{l,t}^{k}$ is evaluated based first on the highest car type, which is normally the most expensive car type. $S_{l,t}^{k,a}$ is evaluated secondly, and $X_{l,l',t}^{k,a}$ is evaluated last. However, $I_{l,t}^{k,a}$ is not included. The change of $I_{l,t}^{k,a}$ will affect at least the inventory balance constraints in two different seasonal periods and make the structure of better neighbors complicated. Furthermore, each move in $B_{l,t}^{k}, S_{l,t}^{k,a}, X_{l,l',t}^{k,a}$ has forced $I_{l,t}^{k,a}$ to change its value. Since the change of a single variable forces other variables to change and the maximal reduction of the objective is adopted for h, some cases may not happen. Based on several trials, the better neighbors of $X_{l,l',t}^{k,a}+h$ never happen, and the better neighbors of $X_{l,l',t}^{k,a}-h$ normally happen in one case and only 4 times in another case, which affected the objective very minimally. Moreover, more cases of $X_{l,l',t}^{k,a}-h$ will result in an extremely heavy computing burden. Hence, all cases in $X_{l,l',t}^{k,a}+h$ and all cases in $X_{l,l',t}^{k,a}-h$ except one case are removed. Because $X_{l,l',t}^{k,a}$ utilize 98.96% of all integer variables, this removal largely reduces the burden of the computing and almost does not affect the objective.

Based on the previous example, there will be 8×12×200=19,200 constraints for the service level and 8×12×200×12=230,400 constraints for inventory balance. Of all constraints, the service level utilizes 7.69%, and the inventory balance utilizes 92.31%. The structure of better neighbors is designed based on the inventory balance constraints. When the single variable changes, exploiting the inventory balance constraints adjusts other changed variables and finds possible cases, which can maintain the inventory balance

constraints automatically and improve the objective. It will not be necessary to evaluate whether the inventory balance constraints, which utilize 92.31% of all constraints, are violated. Furthermore, in such a neighborhood structure, most of the service level constraints also will be automatically satisfied and no additional evaluation is needed. Only 3 cases out of all 10 cases of better neighbors are needed to examine whether one or two of their service level constraints are satisfied. Such a design structure can save at least 99% of the computing time of constraint evaluations.

5.4 Algorithm Procedure

The process of obtaining a feasible initial solution is introduced in Section 5.4.1.In addition, based on the previous motivation in Section 5.3, the structure of better neighbors and the best-improvement descent local search are proposed in Section 5.4.2 and Section 5.4.3.

5.4.1 Initial Solution

The initial solution is generated for an almost worst case and given based on four basic rules as follows.

- If possible, set $S_{l,t}^{k,a} = 0$, $X_{l,l',t}^{k,a} = 0$
- If possible, set $B_{l,t}^{k} = d_{l,t}^{k}$
- Calculate $I_{l,t}^{k,a}$ based on the inventory balance constraints
- Slightly adjust $B_{l,t}^{k}, S_{l,t}^{k,a}, I_{l,t}^{k,a}, X_{l,l',t}^{k,a}$ if still infeasible

Basically, $X_{l,l',t}^{k,a}$ is set to be 0 except in Step 6. The inventory balance constraints become constraints 5.7 and 5.8.

$$I_{l,(t-1)}^{k,(a-1)} + B_{l,t}^{k} - S_{l,t}^{k,a} = I_{l,t}^{k,a} \qquad \text{for a} = 1, \forall l,k,t \qquad (5.7)$$

$$I_{l,(t-1)}^{k,(a-1)} - S_{l,t}^{k,a} = I_{l,t}^{k,a} \qquad \text{for a} \neq 1, \forall l,k,t \qquad (5.8)$$

The detailed procedure of finding an initial feasible solution is listed in the following steps.

Step 1: For $a=1, t=1\sim(t_{\max}-1), \forall k,l$, let $B_{l,t}^{k}=d_{l,t}^{k}, S_{l,t}^{k,a=1}=0, X_{l,l',t}^{k,a=1}=0, I_{l,t}^{k,a=1}=d_{l,t}^{k}$.

When all are assigned, go to Step 2.

Step 2: For $a=1, t=t_{\max}, \forall k,l$, because $I_{l,t_{\max}}^{k,a=1}$ is known,

if $d_{l,t\max}^{k}>I_{l,t_{\max}}^{k,a=1}$, let $B_{l,t_{\max}}^{k}=d_{l,t_{\max}}^{k}, S_{l,t_{\max}}^{k,a=1}=d_{l,t_{\max}}^{k}-I_{l,t_{\max}}^{k,a=1}$.

if $d_{l,t\max}^{k}\le I_{l,t_{\max}}^{k,a=1}$, let $B_{l,t_{\max}}^{k}=I_{l,t_{\max}}^{k,a=1}, S_{l,t_{\max}}^{k,a=1}=0, X_{l,l',t}^{k,a}=0$.

When all are assigned, go to Step 3.

Step 3: For $a=2\sim a_{\max-1}, t=1\sim t_{\max-1}, \forall k,l$, because $I_{l,t'=0}^{k,a-t}$ is known and $I_{l,t-a+1}^{k,a'=1}$ has been given from case 2,

if $a>t$, let $I_{l,t}^{k,a}=I_{l,t'=0}^{k,a-t}, X_{l,l',t}^{k,a}=0, S_{l,t}^{k,a}=0$.

if $a\le t$, let $I_{l,t}^{k,a}=I_{l,t-a+1}^{k,a'=1}, X_{l,l',t}^{k,a}=0, S_{l,t}^{k,a}=0$.

When all are assigned, go to Step 4.

Step 4: For $a=a_{\max}, \forall k,l,t$, $I_{l,t}^{k,a_{\max}}=0$ is known, let $S_{l,t}^{k,a_{\max}}=I_{l,t-1}^{k,a_{\max-1}}, X_{l,l',t}^{k,a_{\max}}=0$.

When all are assigned, go to Step 5.

Step 5: For $t = t_{\max}, a = 2 \sim a_{\max-1}, \forall k, l,$ because $I_{l,t_{\max}}^{k,a}$ is known,

if $I_{l,t_{\max}}^{k,a} \le I_{l,t_{\max-1}}^{k,a-1}$, let $S_{l,t_{\max}}^{k,a} = I_{l,t_{\max-1}}^{k,a-1} - I_{l,t_{\max}}^{k,a}$, $X_{l,l',t_{\max}}^{k,a} = 0.$

if $I_{l,t_{\max}}^{k,a} > I_{l,t_{\max-1}}^{k,a-1}$, calculate and record $tmp\_S_{l,t_{\max}}^{k,a} = I_{l,t_{\max-1}}^{k,a-1} - I_{l,t_{\max}}^{k,a} < 0.$ Let

$X_{l,t_{\max}}^{k,a} = 0$

When all are assigned, go to Step 6.

Step 6: For any $tmp\_S_{l,t_{\max}}^{k,a}$, find $S_{l',t_{\max}}^{k,a} \ge -tmp\_S_{l,t_{\max}}^{k,a}$.

Let $S_{l,t_{\max}}^{k,a} = 0, S_{l',t_{\max}}^{k,a} = S_{l',t_{\max}}^{k,a} + tmp\_S_{l,t_{\max}}^{k,a}$, $X_{l',l,t_{\max}}^{k,a} = -tmp\_S_{l,t_{\max}}^{k,a}$.

When all are assigned, the feasible initial solution is done.

5.4.2 The Structure of Better Neighbors

The neighborhood structure of this problem includes five types of variable changes based on the change of single variables, $B_{l,t}^{k}, S_{l,t}^{k,a}$, or $X_{l,l',t}^{k,a}$. The better neighbors are obtained from the value of a specific variable adding/subtracting a flexible value *h* and are only adopted for those which have better objectives than the current solution. A flexible value *h* is decided based on the maximal reduction of the objective. Several possible cases are covered for each type of variable changes. In addition, the structure of better neighbors is designed based on the inventory balance constraints. These five types of better neighbors include $B_{l,t}^{k}-h$, $B_{l,t}^{k}+h$, $S_{l,t}^{k,a}-h$, $S_{l,t}^{k,a}+h$, and $X_{l,l',t}^{k,a}-h$. The detailed structure is introduced below.

1) $B_{l,t}^{k}-h$:

The prerequisite for this type of better neighbor is $B_{l,t}^{k}>0$

- Case 1:

  Let $a=1$. For any $l'\neq l$, if the difference of the objective $diff_{obj}=b_{l,t}^{k}-tr_{l',l,t}^{k,a}-b_{l',t}^{k}>0$, then $h=B_{l,t}^{k}$. Let $B_{l,t}^{k}=B_{l,t}^{k}-h, X_{l',l,t}^{k,a}=X_{l',l,t}^{k,a}+h, B_{l',t}^{k}=B_{l',t}^{k}+h$. The constraint representation of this case is

$$B_{l,t}^{k}\Downarrow -S_{l,t}^{k,a}+\sum_{l',l'\neq l}X_{l',l,t}^{k,a}\Uparrow -\sum_{l',l'\neq l}X_{l,l',t}^{k,a}=I_{l,t}^{k,a} \qquad \textit{for } l$$

$$B_{l',t}^{k}\Uparrow -S_{l',t}^{k,a}+\sum_{l,l\neq l'}X_{l,l',t}^{k,a}-\sum_{l,l\neq l'}X_{l',l,t}^{k,a}\Uparrow =I_{l',t}^{k,a} \qquad \textit{for } l'$$

- Case 2:

  Let $a=1$. For any $l' \neq l$, if $S_{l',t}^{k,a} > 0$ and the difference of the objective $diff_{obj} = b_{l,t}^{k} - tr_{l',l,t}^{k,a} - s_{l',t}^{k,a} > 0$, then $h = \min\{ B_{l,t}^{k}, S_{l',t}^{k,a} \}$. Let $B_{l,t}^{k} = B_{l,t}^{k} - h, X_{l',l,t}^{k,a} = X_{l',l,t}^{k,a} + h,$ $S_{l',t}^{k,a} = S_{l',t}^{k,a} + h$. The constraint representation of this case is

$$B_{l,t}^{k} \Downarrow -S_{l,t}^{k,a} + \sum_{l',l' \neq l} X_{l',l,t}^{k,a} \Uparrow - \sum_{l',l' \neq l} X_{l,l',t}^{k,a} = I_{l,t}^{k,a} \qquad \text{for } l$$

$$B_{l',t}^{k} - S_{l',t}^{k,a} \Downarrow + \sum_{l,l \neq l'} X_{l,l',t}^{k,a} - \sum_{l,l \neq l'} X_{l',l,t}^{k,a} \Uparrow = I_{l',t}^{k,a} \qquad \text{for } l'$$

- Case 3:

  Let $a=1$.

  For *any* $l' \neq l$

  For $n = 1$ to $\min\{ a_{\max} - a, t_{\max} - t \}$

  {

  If $S_{l',t+n}^{k,a+n} > 0$, calculate $diff_{obj} = b_{l,t}^{k} - tr_{l',l,t}^{k,a} - s_{l',t+n}^{k,a+n}$. Otherwise, the neighbor is not selected.

  For $n' = 0$ to $(n-1)$

  {

  If $I_{l',t+n'}^{k,a+n'} > 0$, calculate $diff_{obj} = diff_{obj} + inv_{l',t+n'}^{k,a+n'}$. Otherwise, the neighbor is not selected.

  }

  If $diff_{obj} > 0$, then $h = \min\{ B_{l,t}^{k}, S_{l',t+n}^{k,a+n}, I_{l',t+n'}^{k,a+n'} \}, n' = 0 \sim (n-1)$.

  Let $B_{l,t}^{k} = B_{l,t}^{k} - h, S_{l',t+n}^{k,a+n} = S_{l',t+n}^{k,a+n} - h, X_{l',l,t}^{k,a} = X_{l',l,t}^{k,a} + h,$.

  $I_{l',t+n'}^{k,a+n'} = I_{l',t+n'}^{k,a+n'} - h$ *for* $n' = 0 \sim (n-1)$.

  Otherwise, the neighbor is not selected.

  }

The constraint representation of this case is

$$B_{l,t}^{k} \Downarrow -S_{l,t}^{k,a} + \sum_{l^{'},l^{'}\neq l} X_{l^{'},l,t}^{k,a} \Uparrow - \sum_{l^{'},l^{'}\neq l} X_{l,l^{'},t}^{k,a} = I_{l,t}^{k,a} \qquad for\ a=1,l$$

$$B_{l',t}^{k} - S_{l',t}^{k,a} + \sum_{l,l\neq l'} X_{l,l',t}^{k,a} - \sum_{l,l\neq l'} X_{l',l,t}^{k,a} \Uparrow = I_{l',t}^{k,a} \Downarrow \qquad for\ a=1,l'$$

$$I_{l',t}^{k,a} \Downarrow -S_{l',t+1}^{k,a+1} + \sum_{l,l\neq l'} X_{l,l',t+1}^{k,a+1} - \sum_{l,l\neq l'} X_{l',l,t+1}^{k,a+1} = I_{l',t+1}^{k,a+1} \Downarrow$$

$$I_{l',t+1}^{k,a+1} \Downarrow -S_{l',t+2}^{k,a+2} \Downarrow + \sum_{l,l\neq l'} X_{l,l',t+2}^{k,a+2} - \sum_{l,l\neq l'} X_{l',l,t+2}^{k,a+2} = I_{l',t+2}^{k,a+2}$$

2) $B_{l,t}^{k} + h$

- Case 1:

  Let $a=1$. For any $l'\neq l$, if $B_{l',t}^{k} > 0$ and the difference of the objective $diff_{obj} = b_{l',t}^{k} - tr_{l,l',t}^{k,a} - b_{l,t}^{k} > 0$, then $h = B_{l',t}^{k}$. Let $B_{l,t}^{k} = B_{l,t}^{k} + h, X_{l,l',t}^{k,a} = X_{l,l',t}^{k,a} + h,$ $B_{l',t}^{k} = B_{l',t}^{k} - h$. The constraint representation of this case is

$$B_{l,t}^{k} \Uparrow -S_{l,t}^{k,a} + \sum_{l^{'},l^{'}\neq l} X_{l^{'},l,t}^{k,a} - \sum_{l^{'},l^{'}\neq l} X_{l,l^{'},t}^{k,a} \Uparrow = I_{l,t}^{k,a} \qquad for\ l$$

$$B_{l',t}^{k} \Downarrow -S_{l',t}^{k,a} + \sum_{l,l\neq l'} X_{l,l',t}^{k,a} \Uparrow - \sum_{l,l\neq l'} X_{l',l,t}^{k,a} = I_{l',t}^{k,a} \qquad for\ l'$$

3) $S_{l,t}^{k,a} - h$

The prerequisite for this type of better neighbor is $S_{l,t}^{k,a} > 0$

- Case 1:

  For any $l'\neq l$, if $a=1$, $B_{l',t}^{k} > 0$ and the difference of the objective $diff_{obj} = b_{l',t}^{k} - tr_{l,l',t}^{k,a} - s_{l,t}^{k,a} > 0$, then $h = \min\{B_{l',t}^{k}, S_{l,t}^{k,a}\}$. Let $S_{l,t}^{k,a} = S_{l,t}^{k,a} - h, X_{l,l',t}^{k,a} = X_{l,l',t}^{k,a} + h,$ $B_{l',t}^{k} = B_{l',t}^{k} - h$. The constraint representation of this case is

$$B_{l,t}^{k} - S_{l,t}^{k,a} \Downarrow + \sum_{l^{'},l^{'}\neq l} X_{l^{'},l,t}^{k,a} - \sum_{l^{'},l^{'}\neq l} X_{l,l^{'},t}^{k,a} \Uparrow = I_{l,t}^{k,a} \qquad \textit{for } l$$

$$B_{l',t}^{k} \Downarrow - S_{l',t}^{k,a} + \sum_{l,l\neq l'} X_{l,l',t}^{k,a} \Uparrow - \sum_{l,l\neq l'} X_{l',l,t}^{k,a} = I_{l',t}^{k,a} \qquad \textit{for } l'$$

- Case 2:

  For any $l' \neq l$, if $a \neq 1, a > t$, and the difference of the objective $diff_{obj} = s_{l',t}^{k,a} - -tr_{l,l',t}^{k,a} - s_{l,t}^{k,a} > 0$, then $h = S_{l,t}^{k,a}$. Let $S_{l,t}^{k,a} = S_{l,t}^{k,a} - h,\ X_{l,l',t}^{k,a} = X_{l,l',t}^{k,a} + h,\ S_{l',t}^{k,a} = S_{l',t}^{k,a} + h$.

  The constraint representation of this case is

$$I_{l,t-1}^{k,a-1} - S_{l,t}^{k,a} \Downarrow + \sum_{l^{'},l^{'}\neq l} X_{l^{'},l,t}^{k,a} - \sum_{l^{'},l^{'}\neq l} X_{l,l^{'},t}^{k,a} \Uparrow = I_{l,t}^{k,a} \qquad \textit{for } l$$

$$I_{l',t-1}^{k,a-1} - S_{l',t}^{k,a} \Uparrow + \sum_{l,l\neq l'} X_{l,l',t}^{k,a} \Uparrow - \sum_{l,l\neq l'} X_{l',l,t}^{k,a} = I_{l',t}^{k,a} \qquad \textit{for } l'$$

4) $S_{l,t}^{k,a} + h$

- Case 1:

  For $n = 1$ to $\min\{a_{\max} - a, t_{\max} - t\}$

  {

  If $S_{l,t+n}^{k,a+n} > 0$, calculate $diff_{obj} = s_{l,t}^{k,a} - s_{l,t+n}^{k,a+n}$. Otherwise, the neighbor is not selected.

  For $n' = 0$ to $(n-1)$

  {

  If $I_{l,t+n'}^{k,a+n'} > 0$, calculate $diff_{obj} = diff_{obj} + inv_{l,t+n'}^{k,a+n'}$. Otherwise, the neighbor is not selected.

  }

  If $diff_{obj} > 0$, then $h = \min\{S_{l,t+n}^{k,a+n}, I_{l,t+n'}^{k,a+n'}\}, n' = 0 \sim (n-1)$.

  Let $S_{l,t}^{k,a} = S_{l,t}^{k,a} + h, S_{l,t+n}^{k,a+n} = S_{l,t+n}^{k,a+n} - h,$

  $I_{l,t+n'}^{k,a+n'} = I_{l,t+n'}^{k,a+n'} - h$ *for* $n' = 0 \sim (n-1)$.

  Otherwise, the neighbor is not selected.

}

The constraint representation of this case is

$$B_{l,t}^{k} / I_{l,t-1}^{k,a-1} - S_{l,t}^{k,a} \Uparrow + \sum_{l^{'},l^{'} \neq l} X_{l^{'},l,t}^{k,a} - \sum_{l^{'},l^{'} \neq l} X_{l,l^{'},t}^{k,a} = I_{l,t}^{k,a} \Downarrow$$

$$I_{l,t}^{k,a} \Downarrow - S_{l,t+1}^{k,a+1} \Downarrow + \sum_{l',l' \neq l} X_{l',l,t+1}^{k,a+1} - \sum_{l',l' \neq l} X_{l,l',t+1}^{k,a+1} = I_{l,t+1}^{k,a+1}$$

- Case 2:

For any $l' \neq l$

For $n = 1$ to $\min\{ a_{\max} - a, t_{\max} - t \}$

{

If $S_{l',t+n}^{k,a+n} > 0$, calculate $diff_{obj} = s_{l,t}^{k,a} - tr_{l',l,t}^{k,a} - s_{l',t+n}^{k,a+n}$. Otherwise, the neighbor is not selected.

For $n' = 0$ to $(n-1)$

{

If $I_{l',t+n'}^{k,a+n'} > 0$, calculate $diff_{obj} = diff_{obj} + inv_{l',t+n'}^{k,a+n'}$. Otherwise, the neighbor is not selected.

}

If $diff_{obj} > 0$, then $h = \min\{ S_{l',t+n}^{k,a+n}, I_{l',t+n'}^{k,a+n'} \}, n' = 0 \sim (n-1)$.

Let $S_{l,t}^{k,a} = S_{l,t}^{k,a} + h, X_{l',l,t}^{k,a} = X_{l',l,t}^{k,a} + h, S_{l',t+n}^{k,a+n} = S_{l',t+n}^{k,a+n} - h,$

$I_{l',t+n'}^{k,a+n'} = I_{l',t+n'}^{k,a+n'} - h$ *for* $n' = 0 \sim (n-1)$.

Otherwise, the neighbor is not selected.

}

The constraint representation of this case is

$$B_{l,t}^{k} / I_{l,t-1}^{k,a-1} - S_{l,t}^{k,a} \Uparrow + \sum_{l^{'},l^{'} \neq l} X_{l^{'},l,t}^{k,a} \Uparrow - \sum_{l^{'},l^{'} \neq l} X_{l,l^{'},t}^{k,a} = I_{l,t}^{k,a}$$

$$B_{l',t}^{k} / I_{l,t-1}^{k,a-1} - S_{l,t}^{k,a} + \sum_{l,l \neq l'} X_{l,l',t}^{k,a} - \sum_{l,l \neq l'} X_{l',l,t}^{k,a} \Uparrow = I_{l',t}^{k,a} \Downarrow$$

$$I_{l',t}^{k,a} \Downarrow - S_{l',t+1}^{k,a+1} \Downarrow + \sum_{l,l \neq l'} X_{l,l',t+1}^{k,a+1} - \sum_{l,l \neq l'} X_{l',l,t+1}^{k,a+1} = I_{l',t+1}^{k,a+1}$$

- Case 3:

  For any $l' \neq l$, if $S_{l',t}^{k,a} > 0$ and the difference of the objective $diff_{obj} = s_{l,t}^{k,a} - tr_{l',l,t}^{k,a} - s_{l',t}^{k,a} > 0$, then $h = S_{l',t}^{k,a}$. Let $S_{l,t}^{k,a} = S_{l,t}^{k,a} + h$, $X_{l',l,t}^{k,a} = X_{l',l,t}^{k,a} + h$, $S_{l',t}^{k,a} = S_{l',t}^{k,a} - h$.

  The constraint representation of this case is

$$B_{l,t}^{k} / I_{l,t-1}^{k,a-1} - S_{l,t}^{k,a} \Uparrow + \sum_{l',l' \neq l} X_{l',l,t}^{k,a} \Uparrow - \sum_{l',l' \neq l} X_{l,l',t}^{k,a} = I_{l,t}^{k,a} \qquad \textit{for } l$$

$$B_{l',t}^{k} / I_{l',t-1}^{k,a-1} - S_{l',t}^{k,a} \Downarrow + \sum_{l,l \neq l'} X_{l,l',t}^{k,a} - \sum_{l,l \neq l'} X_{l',l,t}^{k,a} \Uparrow = I_{l',t}^{k,a} \qquad \textit{for } l'$$

5) $X_{l,l',t}^{k,a} - h$

The prerequisite for this type of better neighbor is $X_{l,l',t}^{k,a} > 0$

- Case 1:

  For any $l' \neq l$

  {

  If $X_{l',l,t}^{k,a} > 0$, calculate $diff_{obj} = tr_{l',l,t}^{k,a} + tr_{l,l',t}^{k,a}$. Otherwise, the neighbor is not selected.

  If $X_{l',l,t}^{k,a} > X_{l,l',t}^{k,a}$, $h = X_{l,l',t}^{k,a}$. Otherwise, $h = X_{l',l,t}^{k,a}$.

  Let $X_{l,l',t}^{k,a} = X_{l,l',t}^{k,a} - h$, $X_{l',l,t}^{k,a} = X_{l',l,t}^{k,a} - h$.

  }

  The constraint representation of this case is

$$B_{l,t}^{k} / I_{l,t-1}^{k,a-1} - S_{l,t}^{k,a} + \sum_{l',l' \neq l} X_{l',l,t}^{k,a} \Downarrow - \sum_{l',l' \neq l} X_{l,l',t}^{k,a} \Downarrow = I_{l,t}^{k,a} \qquad \textit{for } l$$

$$B_{l',t}^{k} / I_{l',t-1}^{k,a-1} - S_{l',t}^{k,a} + \sum_{l,l \neq l'} X_{l,l',t}^{k,a} \Downarrow - \sum_{l,l \neq l'} X_{l',l,t}^{k,a} \Downarrow = I_{l',t}^{k,a} \qquad \textit{for } l'$$

5.4.3 Best-Improvement Descent Local Search

Based on the structure of better neighbors, a simple algorithm called best-improvement descent local search is proposed and introduced.

Step 1: Let $Iter_{pre\_loop}=0$ and $Iter_{current}=0$. Go to Step 2.

Step 2:

For car type $k=k_{max}$ to 1

For pool $l=1$ to l_{max}

For period $t=1$ to t_{max}

Step 2.1: Calculate $diff_{obj}$ for all better neighbors i of $B_{l,t}^{k}-h, B_{l,t}^{k}+h$.

Step 2.2: Rank these neighbors i in descending order of $diff_{obj}$ and obtain a list of ranked neighbors, denoted as LB_i.

Step 2.3: Pick a neighbor i^* from LB_i and check if constraints (5.2) are satisfied. If yes, move to i^*, $Iter_{current}= Iter_{current}+1$, and go to Step 2.1; if not, choose the next available from LB_i until all neighbors are checked.

End

End

End

Go to Step 3.

Step 3:

For car type $k=k_{max}$ to 1

For pool $l=1$ to l_{max}

For period $t=1$ to t_{max}

For car age $a=1$ to a_{max}

Step 3.1: Calculate $diff_{obj}$ for all better neighbors i of $S_{l,t}^{k,a} - h, S_{l,t}^{k,a} + h$.

Step 3.2: Rank these neighbors i in descending order of $diff_{obj}$ and obtain a list of ranked neighbors, denoted as LS_i.

Step 3.3: Pick a neighbor $i*$ from LS_i and check if constraints (5.2) are satisfied. If yes, move to $i*$, $Iter_{current}$= $Iter_{current}$ +1, and go to Step 3.1; if not, choose the next available from LS_i until all neighbors are checked.

End

End

End

End

Go to Step 4.

Step 4:

For car type $k=k_{max}$ to 1

For pool $l=1$ to l_{max}

For period $t=1$ to t_{max}

For car age $a=1$ to a_{max}

For pool $l'=1$ to l_{max}

Step 4.1: Calculate $diff_{obj}$ for all better neighbors i of $X_{l,t}^{k,a} - h$.

Step 4.2: Rank these neighbors i in descending order of $diff_{obj}$ and obtain a list of ranked neighbors, denoted as LX_i.

Step 4.3: Pick a neighbor $i*$ from LX_i and check if constraints (5.2)

are satisfied. If yes, move to i^*, $Iter_{current} = Iter_{current} + 1$, and go to Step 4.1; if not, choose the next available from LX_i until all neighbors are checked.

End

End

End

End

End

Go to Step 5.

Step 5: If $Iter_{pre\_loop} \neq Iter_{current}$, let $Iter_{pre\_loop} = Iter_{current}$ and go to Step 2; otherwise, the whole algorithm is finished.

5.5 Computing Results

This section focuses on an experimental design generated by three types of experimental factors including: the number of car types, the number of pools, and the number of seasonal periods/car ages. Section 5.5.1 introduces the parameter setting and the meaning of the parameters. In section 5.5.2, the setting of factor levels and the two parts of the experiment are described, including small scale problems and large scale problems. The software and the computer equipment for conducting this experiment are described in Section 5.5.3 In Section 5.5.4, experimental results are analyzed and explained.

5.5.1 Parameter Settings

The parameter settings in inter-pool moves and asset replacement are chosen based on the website data of Auto Rental News or the assumptions of making the problems feasible, and are as described below.

- ✧ **Demand**: Because the demand for small-sized cars is normally higher than that of the larger models at the same location, the demands for different sizes have the characteristic of this dependence. Hence, the demand for car type 1 in each season at each pool is generated by multiplying a random demand level, generated from a uniform distribution (40, 60), by a random time ratio, generated from a uniform distribution (0.5, 1.5). As for the demands for car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (0.6, 1.2) for each car type. The demand for car type 2 is then generated by multiplying the demand for car type 1 by the ratio for car

type 2. The demands for the higher car types are generated by multiplying the demand for the previous car type by the respective ratio. The demand is rounded off to an integer.

- ✧ **Unit Price of Buying A Car:** The unit price of buying a car with a car type 1 in each season at each pool is generated by multiplying a random unit price level, generated from a uniform distribution (12000, 13000), by a random time ratio generated from a uniform distribution (0.9, 1.1). As for the unit buying price of car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (2.0, 2.1) for each car type. The unit buying price of car type 2 is then generated by multiplying the demand for car type 1 by the ratio for car type 2. The unit buying prices of the higher car types are generated by multiplying the demand for the previous car type by the respective ratio. The unit price of buying a car is rounded off to four decimal places.
- ✧ **Unit Price of Selling A Car:** The unit price of selling a car for each car type in each season at each pool is generated by multiplying its unit price of buying a car by a random car age ratio generated from a uniform distribution (0.5, 0.6). The unit price of selling a car is rounded off to four decimal places.
- ✧ **Unit Inventory Cost:** The unit inventory cost for car type 1, car age 1 in each season at each pool is generated by multiplying a random unit inventory level, generated from a uniform distribution (15, 70), by a random time ratio generated from a uniform distribution (0.8, 1.2). As for the unit inventory cost for car type 2 and the higher car types, a type ratio value 1.1 is generated. The type ratio for car type k equals $1.1^{(k-1)}$. The unit inventory cost for car age 2 and higher, an age ratio value 1.1 is generated. The car age ratio for car age a equals $1.1^{(k-1)}$. The unit inventory cost of car type k, car age a is then generated by multiplying the unit inventory cost of car type 1, car age 1 by the

type ratio for car type k and the age ratio for car age a. The unit inventory cost is rounded off to four decimal places.

- **Transportation Cost:** The unit transportation cost from pool l to pool l' for car type 1, car age 1 in each season at each pool is generated by multiplying a random unit transportation cost level, generated from a uniform distribution (1, 3), by a random time ratio generated from a uniform distribution (0.9, 1.1). As for the unit transportation cost for car type 2 and the higher car types, a type ratio value 1.1 is generated. The type ratio for car type k equals $1.1^{(k-1)}$. The unit transportation cost for car age 2 and higher car ages, a age ratio value 1.05 is generated. The car age ratio for car age a equals $1.05^{(k-1)}$. The unit transportation cost from pool l to pool l' for car type k, car age a is then generated by multiplying the unit transportation cost from pool l to pool l' for car type 1, car age 1 by the type ratio for car type k and the age ratio for car age a. The transportation cost is rounded off to four decimal places.
- **Initial Inventory:** The initial inventory of car type 1 in each car age within each pool is generated by multiplying a random initial inventory level, generated from a uniform distribution (40, 60), by a random time ratio, generated from a uniform distribution (0.8, 1.2). As for the initial inventory of car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (0.6, 1.2) for each car type. The initial inventory of car type 2 is then generated by multiplying the initial demand for car type 1 by the ratio for car type 2. The initial demands for the higher car types are generated by multiplying the demand for the previous car type by the respective ratio. The initial inventory is rounded off to an integer.

- ✧ **Final Inventory**: The parameter setting for the final inventory is the same as the initial inventory.
- ✧ **Service Ratio:** In this problem, the service ratio is set to be 1, which means all demands need to be satisfied.

5.5.2 Factor Levels

Two parts of the experimental designs are exploited in inter-pool moves and asset replacement. In part 1, small scale problems are tested and compared to the branch-and-bound algorithm, to determine which factor can significantly affect the solution quality and the algorithm time. In part 2, the algorithm time is tested and compared in large scale problems.

Based on the experimental design in part 1, the following 3 factors are tested to determine if the solution quality and the computing time are significant affected.

- ✧ **Number of Pools:** Three factor levels are set to be 10, 20, and 30, which are the maximal number of pools available to solve by the branch-and-bound through LINGO Software.
- ✧ **Number of Car Types:** Three factor levels are set to be 4, 5, and 6.
- ✧ **(Number of Car Ages, Number of Seasonal Periods):** Three factor levels are set to be (4, 4), (5, 5), and (6, 6).

In part 2, the large scale problems are tested to understand how large of a problem size this algorithm can solve and how long it will take. The number of car types and the number

of car ages/seasonal periods are fixed to be 8 and 12. Only the numbers of pools are tested for 6, 9, 12, 50, 100, 150, and 200.

5.5.3 Experimental Platform

This experiment uses the software Visual C++ to compile the computer coding of the best-improvement descent local search and uses the optimization software LINGO 9.0 to solve the optimal solution. The computer equipment for conducting this experiment includes an Intel Core 2 Duo E7400 2.80 GHz CPU and 6 GB memory.

5.5.4 Experimental Analysis

The experiment is divided into two parts. Three types of experimental factors are utilized to test which factors affect the computing time and the solution quality in part 1. Each factor contains three factor levels and each factor level uses three replications based on different random seeds. Hence, 3 factor levels of the number of pools $\times 3$ factor levels of the number of car types $\times$ 3 factor levels of the number of car ages/seasonal periods $\times$ 3 random seeds $=$ 81 independent trials in part 1.

In part 2, the number of pools is the only experimental factor and it contains seven factor levels. Hence, 7 factor levels of the number of pools $\times$ 3 random seeds = 21 independent trials in part 2.

The illustration of the Analysis of Variance (ANOVA) and Tukey test conducted can be referenced in Section 4.5.4

5.5.4.1 Impact Analysis on Experimental Factors versus Solution Gap

The ANOVA table on three factors versus the solution gap is shown in Table 5.1. Observe in Table 5.1 that only the *p*-value of the number of ages/seasons is less than 0.05. This means that the factor effect of the number of ages/seasons is significant. Hence, the number of ages/seasons can affect the solution quality of the algorithm. The original results of this part can be reference in Appendix E.

Table 5.1. ANOVA table on three factors versus solution gap

Source	DF	SSE	MSE	F value	p value
Car Types	2	0.0002070	0.0001035	1.41	0.251
Ages/Seasons	2	0.0092451	0.0046226	63.00	0.000
Pools	2	0.0000083	0.0000042	0.06	0.945
Error	74	0.0054295	0.0000734		
Total	80	0.0148899			

Tukey's test on different numbers of car ages/seasonal periods is represented in Table 5.2. The original results of Tukey's test on the number of ages/seasons are represented in Appendix F. In Table 5.2, there are significant differences in the solution gap among the numbers of car ages/seasonal periods. The average solution gaps of car ages/seasonal periods 4, 5, and 6 are 3.34%, 1.47%, and 0.82%, respectively. The larger the number of ages/seasons, the smaller the solution gap.

Table 5.2. Tukey test on different numbers of ages/seasons

Ages/Seasons	# of Trials	Group (solution gap)		
		A	B	C
4	27	3.34%		
5	27		1.47%	
6	27			0.82%

The figures of average solution gap on different levels of these three experimental factors are summarized as in Figure 5.2.

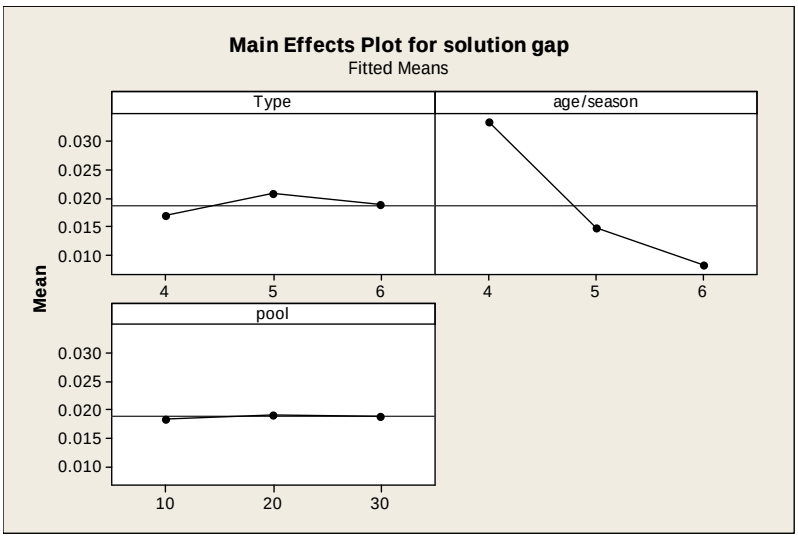

Figure 5.2. Average solution gaps on different levels of experimental factors

5.5.4.2 Impact Analysis on Experimental Factors versus Computing Time

In this section, two ANOVA tables on three factors versus the algorithm time and the time solved by the branch-and-bound method are represented in Table 5.3 and Table 5.4. The original results of this part can be referenced in Appendix E. Note from Table 5.3 and Table 5.4 that all p-values of these three factors are less than 0.05. That means that the effects of these three factors are all significant. Hence, the number of car types, the number of car ages/seasonal periods, and the number of pools can affect the computing time of the algorithm and the computing time of the branch-and-bound method.

Table 5.3. ANOVA table on three factors versus the algorithm time

Source	DF	SSE	MSE	F value	p value
Car Types	2	0.22465	0.11233	67.45	0.000
Ages/Seasons	2	0.18063	0.09032	54.23	0.000
Pools	2	1.45017	0.72508	435.38	0.000
Error	74	0.12324	0.00167		
Total	80	1.97869			

Table 5.4. ANOVA table on three factors versus the time solved by the branch- and-bound method

Source	DF	SSE	MSE	F value	p value
Car Types	2	59959	29979	12.14	0.000
Ages/Seasons	2	122894	61447	24.88	0.000
Pools	2	440060	220030	89.08	0.000
Error	74	182755	2470		
Total	80	805668			

Next, Tukey's test is conducted to compare different factor levels on each experimental factor. The original results of Tukey's test are represented in Appendix G.

Tukey's tests for the algorithm time and the time solved by the branch-and-bound method on different numbers of car types are represented in Table 5.5 and Table 5.6. In Table 5.5, there are significant differences in the algorithm time among different numbers of car types. The fewer the number of car types, the less the algorithm time. In Table 5.6, there seems not to be any difference in the time solved by the branch-and-bound method between the numbers of car type 4 and 5. However, there are significant differences in the time solved by the branch-and-bound method between the numbers of car types, 4 and 6, or, 5 and 6. The time solved by the branch-and-bound method is normally 200 to 300 times more than the algorithm time.

Table 5.5. Tukey test of the algorithm time on different numbers of car types

Car Types	# of Trials	Group (algorithm time(sec))		
		A	B	C
4	27	0.247		
5	27		0.312	
6	27			0.376

Table 5.6. Tukey test of the time solved by the branch-and-bound method on different numbers of car types

Car Types	# of Trials	Group (LINGO time(sec))	
		A	B
4	27	50	
5	27	78	
6	27		116

Tukey's tests for the algorithm time and the time solved by the branch-and-bound method on different numbers of car ages/seasonal periods are represented in Table 5.7 and Table 5.8. In Table 5.7 and Table 5.8, there are significant differences in the algorithm time and in the time solved by the branch-and-bound among different numbers of car ages/seasonal periods, respectively. The fewer the number of car ages/seasonal periods, the less the algorithm time or the less the time solved by the branch-and-bound method. The time solved by the branch-and-bound method is normally 150 to 350 times more than the algorithm time.

Table 5.7. Tukey test of the algorithm time on different numbers of car ages/seasonal periods

Ages/Seasons	# of Trials	Group (algorithm time(sec))		
		A	B	C
4	27	0.254		
5	27		0.311	
6	27			0.370

Table 5.8. Tukey test of the time solved by the branch-and-bound method on different numbers of car ages/seasonal periods

Ages/Seasons	# of Trials	Group (LINGO time(sec))		
		A	B	C
4	27	37		
5	27		75	
6	27			132

Tukey's tests for the algorithm time and the time solved by the branch-and-bound method on different numbers of pools are represented in Table 5.9 and Table 5.10. In Table 5.9 and Table 5.10, there are significant differences in the algorithm time and in the time solved by the branch-and-bound among different numbers of pools, respectively. The fewer the number of pools, the less the algorithm time or the less the time solved by the branch-and-bound method. The time solved by the branch-and-bound method is normally 50 to 380 times more than the algorithm time.

Table 5.9. Tukey test of the algorithm time on different numbers of pools

Pools	# of Trials	Group (algorithm time(sec))		
		A	B	C
10	27	0.152		
20	27		0.303	
30	27			0.479

Table 5.10. Tukey test of the time solved by the branch-and-bound method on different numbers of pools

Pools	# of Trials	Group (algorithm time(sec))		
		A	B	C
10	27	7.5		
20	27		54.8	
30	27			182.0

The figures of average algorithm time and average time solved by the branch-and-bound method on different levels of these three experimental factors are summarized as in Figure 5.3 and Figure 5.4.

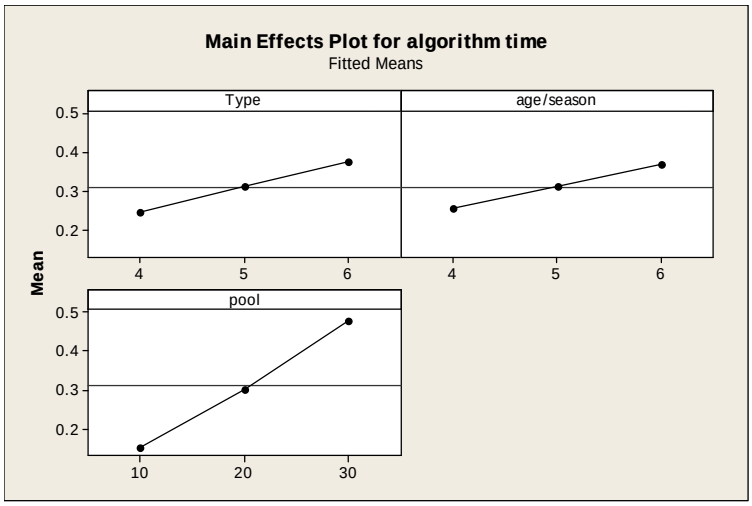

Figure 5.3. Average algorithm times on different levels of experimental factors

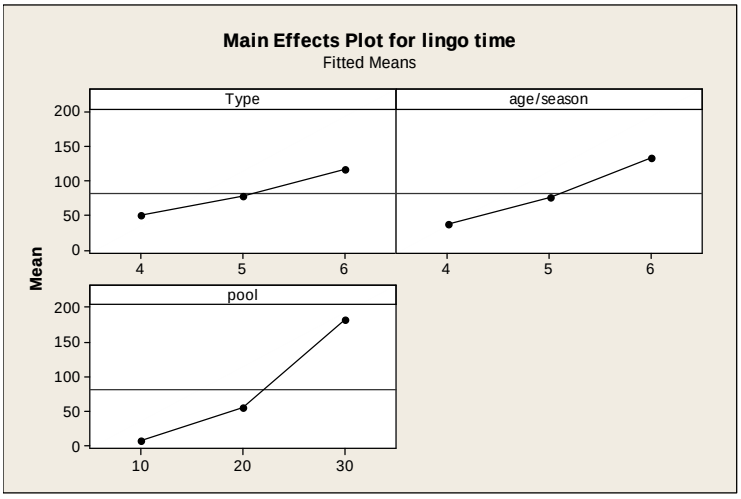

Figure 5.4. Average time solved by the branch-and-bound method

on different levels of experimental factors

5.5.4.3 Impact Analysis on Large Number of pools versus Computing Time

The experiment of part 2 is the impact analysis on the large number of pools versus algorithm time. The number of pools is the most important experimental factor in this problem because it affects how large of a problem size this algorithm can solve. Table 5.11 represents the algorithm time and the number of integer variables on different numbers of pools and its trend chart is presented in Figure 5.5. The branch-and-bound method has the problem of memory overflow in the large case and cannot obtain their results. The largest case, 200, covers 46 million integer variables and 250 thousand constraints and the algorithm takes only 38.6 seconds to solve.

Table 5.11. The algorithm time on the large number of pools

EX	# of Pools	# of Integer Var	# of Constraints	Seed #	Algorithm		Branch-and-Bound		solution gap(%)	Avg solution gap (%)
					Time (sec)	Avg Time(sec)	Time (sec)	Avg Time(sec)		
1	6	48,960	7,489	20	0.28	0.270	55	55.0	0.23%	0.20%
2				40	0.265		54		0.14%	
3				60	0.265		56		0.22%	
4	9	104,544	11,233	20	0.422	0.432	195	194.3	0.15%	0.14%
5				40	0.437		195		0.12%	
6				60	0.437		193		0.14%	
7	12	180,864	14,977	20	0.608	0.608	452	448.7	0.24%	0.20%
8				40	0.578		442		0.19%	
9				60	0.639		452		0.17%	
10	50	2,942,400	62,401	20	3.541	3.458	Memory Overflow			
11				40	3.339					
12				60	3.494					
13	100	11,644,800	124,801	20	9.375	9.968				
14				40	10.686					
15				60	9.843					
16	150	26,107,200	187,201	20	19.781	20.020				
17				40	21.512					
18				60	18.767					
19	200	46,329,600	249,601	20	38.485	38.631				
20				40	38.907					
21				60	38.501					

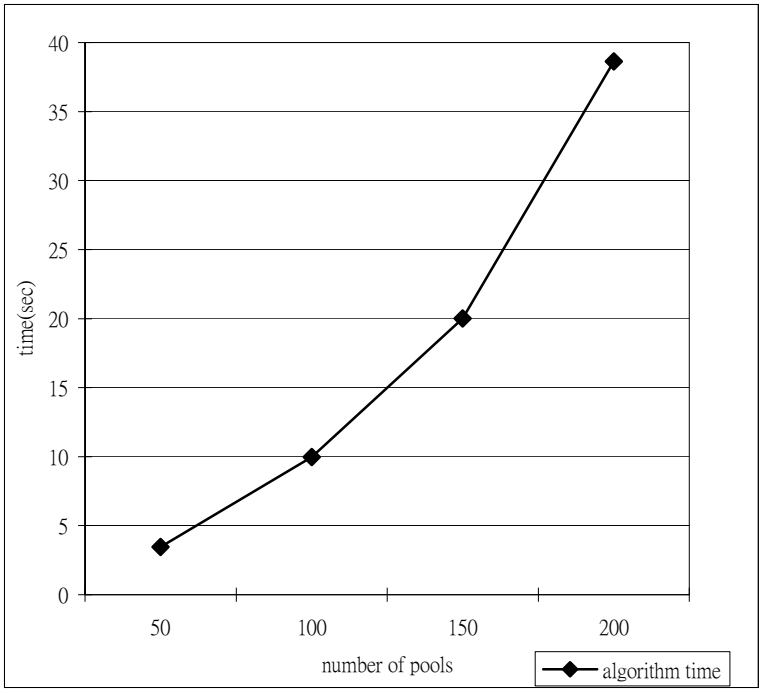

Figure 5.5. The trend chart of algorithm time on the large number of pools

Figure 5.6 is the trend chart of the algorithm time in example 25 (The number of pools is 200 and seed # is 20). The solution gap is calculated by $\frac{Z_{current} - Z_{final}}{Z_{final}} \times 100\%$ and it decreases rapidly from 45% to 10% in the first 5 seconds. Several Steep points as the dashed circles fall at the start of Step 2, Step 3, and Step 4 of the best-improvement descent local search. The solution gap decreases to almost 0% after 30 seconds.

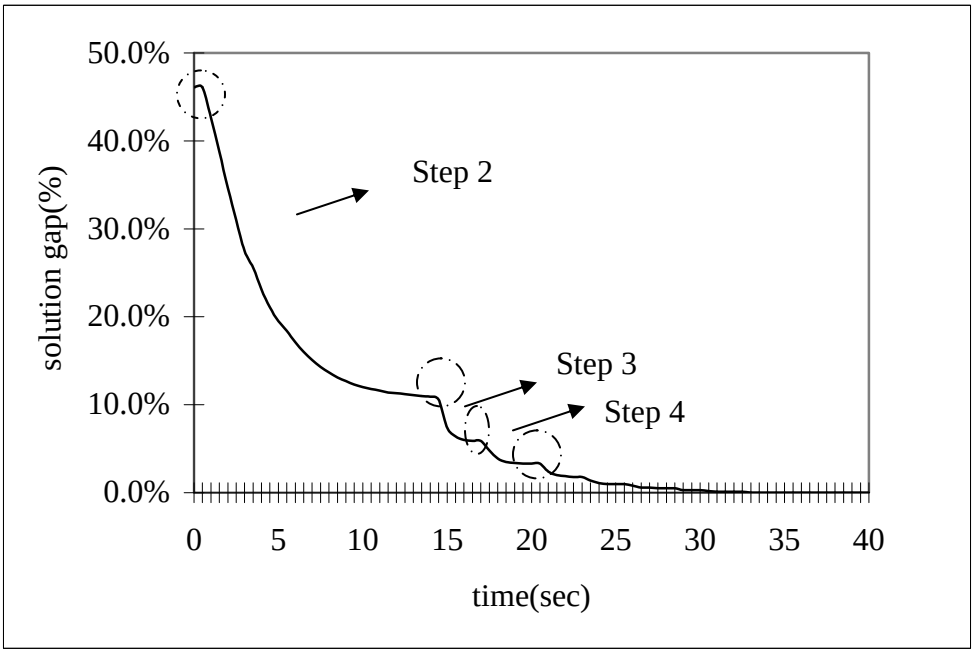

Figure 5.6. The trend chart of the algorithm time in Example 25

5.6 Concluding Remarks

In this chapter, an integer model of inter-pool moves and asset replacement was introduced and a best-improvement descent local search with the structure of better neighbors was proposed and validated. The structure of better neighbors exploits the concept of the maximal objective reduction to decide a flexible value h in the neighbors $Var \pm h$. Compared to a fixed value h, this reduces the number of iterations in the same type of neighbor. In addition, the better neighbors exploit the inventory balance constraints to adjust other variables and largely reduce the computing time of evaluating the constraints. In this algorithm, $B_{l,t}^{k}$, $S_{l,t}^{k,a}$, and $X_{l,l',t}^{k,a}$ are evaluated in the order of their impacts on the objective. This design enables the algorithm to quickly decrease the objective early in the computing process.

Based on the numerical results, the number of car ages/seasonal periods can affect the solution gap but the number of car types and the number of pools do not. The larger the number of car ages/seasonal periods, the smaller the solution gap. The solution gap normally falls below 4%. If the problem size becomes larger, the solution gap normally becomes smaller. This result demonstrates that the best-improvement descent local search will have better results as the problem size becomes larger. While observing the computing times of the algorithm and the branch-and-bound method, all three experimental factors can affect the computing times in these two methods. The average time solved by the branch-and-bound method is about 50 to 380 times more than the algorithm time. The largest problem tested in this algorithm contains 200 pools, 46 million integer variables, and 250 thousand constraints and the best-improvement descent local search only takes 38.6 seconds to solve. These

numerical results show that the best-improvement descent local search has very good performance and can obtain a near-optimal solution in an extremely short time.

CHAPTER 6

DEMAND ALLOCATION AND

EMPTY FLOW REDISTRIBUTION

6.1 Problem Formulation

In Chapter 4, all locations are allocated to pool regions and hubs are selected based on the yearly demand. In Chapter 5, seasonal inter-pool moves and asset replacement are implemented. However, it is necessary to address daily planning in the same pool region. The fact that the customers can rent a car and return it either in the same or in a different location leads to the problem of vehicle imbalance, and empty vehicles need to be redistributed to meet demand at individual locations. In the same pool, empty cars can be redistributed the same day. Moreover, the car upgrade policy and service level are considered. The network flow of demand allocation and empty flow redistribution is presented in Figure 6-1.

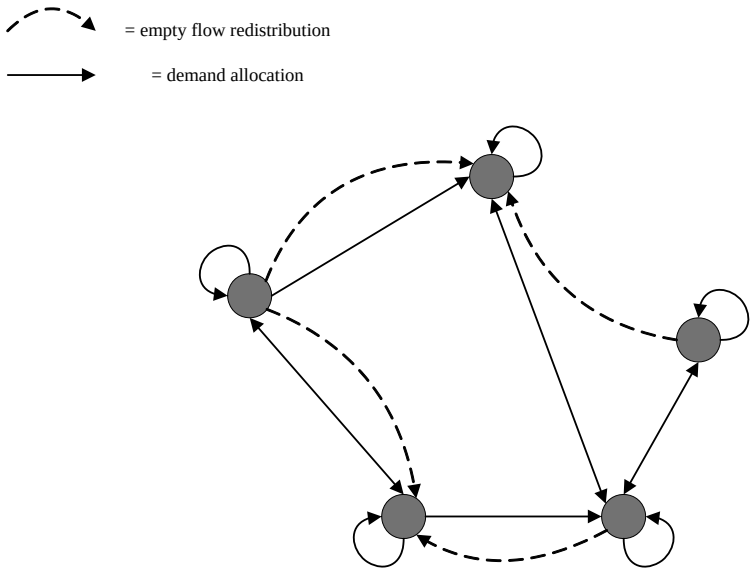

Figure 6.1. Demand allocation and empty flow redistribution

6.2 Mathematical Model

This section introduces a mathematical model to solve the problem of demand allocation and empty flow redistribution. In addition, the upgrade policy and a specific service ratio of satisfying the exact demand are included. This problem is formulated as an integer programming model.

Indices

l = location l
t = time period t
k = car type k
r = rental period r

Parameters

$d_{l,l',t}^{k,r}$ = demand for car type k and rental period r to move from location l to location l' at time period t
$m_{l,t}^{k}$ = unit inventory cost for an idle car of type k at location l at time period t
$v_{l,l',t}^{k,r}$ = operatonal and maintenance cost for a rental car of type k and rental period r from location l to location l' at time period t
$c_{l,l',t}^{k}$ = operational and maintenance cost for an empty car of type k from location l to location l' at time period t
α^{k} = ratio of satisfying exact demand for car type k without upgrading

Variables

$X^{k}_{l,l',t}$ = empty flow of car type k from location l to location l' at time period t

$I^{k}_{l,t}$ = inventory of car type k held at location l at the end of time period t

$d^{-k,k',r}_{l,l',t}$ = rental car flow of type k' serving demand for car type k and rental period r from location l to location l' at time period t

The model can be expressed as:

$$Min \quad \sum_{t}\sum_{k}\sum_{l}\sum_{l',l'\neq l} c^{k}_{l,l',t} X^{k}_{l,l',t} + \sum_{t}\sum_{k}\sum_{l}\sum_{l'}\sum_{k'=1}^{k}\sum_{r} v^{k,r}_{l,l',t} d^{-k',k,r}_{l,l',t} + \sum_{t}\sum_{k}\sum_{l} m^{k}_{l,t} I^{k}_{l,t} \qquad (6.1)$$

subject to:

$$\sum_{k_1=k}^{k_{\max}}\sum_{k_2=k_1}^{k_{\max}} d^{-k_1,k_2,r}_{l,l',t} \geq \sum_{k_1=k}^{k_{\max}} d^{k_1,r}_{l,l',t} \qquad \forall l,l',t,r,k \qquad (6.2)$$

$$\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l',l,(t-r)} + \sum_{l',l'\neq l} X^{k}_{l',l,t} + I^{k}_{l,(t-1)} - \sum_{l',l'\neq l} X^{k}_{l,l',t} = \sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l,l',t} + I^{k}_{l,t} \qquad \forall k,l,t \qquad (6.3)$$

$$\sum_{t}\sum_{r}\sum_{l}\sum_{l'} d^{-k,k,r}_{l,l',t} \geq \alpha^{k} \times \sum_{t}\sum_{r}\sum_{l}\sum_{l'} d^{k,r}_{l,l',t} \qquad \forall\ k,l,t,\ 0 \leq \alpha^{k} \leq 1 \qquad (6.4)$$

$X^{k}_{l,l',t}$ are integers $\qquad \forall k,l,l',t,l \neq l' \qquad (6.5)$

$d^{-k,k',r}_{l,l',t}$, $I^{k}_{l,t}$ are integers $\qquad \forall k,k',r,l,l',t \qquad (6.6)$

$X^{k}_{l,l',t=0}$ are given $\qquad \forall k,l,l',t,l \neq l' \qquad (6.7)$

$I^{k}_{l,t=0}$, $d^{-k,k',r}_{l,l',(t-r)\leq 0}$ are given $\qquad \forall k,k',l,l',t,r \qquad (6.8)$

In Objective (6.1), the objective function is to minimize the operational and maintenance cost of empty car flow, rental car flow, and inventory cars. Constraints (6.2) ensure that the demand for car type k plus higher types can always be allocated. Constraints (6.3) indicate that the car type k flow of rental cars, empty cars, and inventory cars at location l in time period t needs to be balanced. $\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l',l,(t-r)}$ and $\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l,l',t}$ designate the non-empty in-flow and out-flow at location l and car type k

at time period t. $\sum_{l',l'\neq l} X^{k}_{l',l,t}$ and $\sum_{l',l'\neq l} X^{k}_{l,l',t}$ designate the empty in-flow and out-flow at location l and car type k at time period t. Constraints (6.4) are used to ensure that at least a specific percentage of the demand for car type k is satisfied by the same car type. Constraints (6.5) and (6.6) are the integrality requirements and the initial levels are given in Constraints (6.7) and (6.8).

6.3 Motivation

It is not difficult to observe that the structure of this mathematical model is similar to the model of inter-pool moves and asset replacement. There are flow balance constraints and capacity constraints in both models. The variable of rental car flow $d^{-k',k,r}_{l,l',t}$ has as many as six parameter indices. That means that if a problem covers 21 rental locations, 21 rental days, 21 planning days, and 10 car types, the number of integer variables will be up to 10.8 millions. It will be very difficult to solve to optimality.

In solving a very large scale integer programming problem, the decomposition method seems not to be effective because, even after decomposing, the number of the variables is still too huge. It also needs a set of effective rules to transfer the LP solution solved by the decomposition method to a feasible IP solution. Instead, if this problem is solved by a neighborhood search, it needs to overcome the problem of the large neighborhood in this problem. In Chapter 5, the design of better neighbors obtains excellent performance. Hence, a suitable neighbor structure based on the concept of better neighbors is proposed to solve this problem.

6.4 Algorithm Procedure

The rule for obtaining a feasible initial integer solution is introduced in Section 6.4.1. In Section 6.4.2, the structure of better neighbors is proposed. A first-improvement descent local search is developed in Section 6.4.3.

6.4.1 Initial Solution

The initial solution is generated based on the following basic rule.

Step 1: Assign $d_{l,l',t}^{-k,k,r} = \alpha^k \times d_{l,l',t}^{k,r}$ for all k, l, l', r, t.

Step 2: For $k = k_{\max} \sim 1$, $\forall r, l, l', t$,

if $k = k_{\max}$, $d_{l,l',t}^{k,k,r} = d_{l,l',t}^{k,r}$.

if $k \neq k_{\max}$, for $k' = (k+1) \sim k_{\max}$, $d_{l,l',t}^{-k,k',r} = (d_{l,l',t}^{k,r} - d_{l,l',t}^{-k,k,r}) / (k_{\max} - k)$.

Step 3: Assign $X_{l,l',t}^{k} = 0$ for all $k, l, l', t, l' \neq l$.

Step 4: $I_{l,t}^{k}$ is obtained from Constraint (6.3).

Step 5: Slightly adjust $I_{l,t}^{k}, X_{l,l',t}^{k}, d_{l,l',t}^{-k,k',r}$ if $I_{l,t}^{k} < 0$.

The detailed procedure of finding an initial feasible solution is listed in the following steps.

Step 1: Assign $d_{l,l',t}^{-k,k,r} = ceil(\alpha^k \times d_{l,l',t}^{k,r})$ for all k,l,l',r,t, ceil function represent the smallest integer value which is larger than $\alpha^k \times d_{l,l',t}^{k,r}$.

Step 2: For $k = k_{\max} \sim 1, \forall r,l,l',t$,

if $k = k_{\max}, d_{l,l',t}^{k,k,r} = d_{l,l',t}^{k,r}$.

if $k \neq k_{\max}$, $left\_capa1 = d_{l,l',t}^{k,r} - d_{l,l',t}^{-k,k,r}$, $left\_type = k_{\max} - k$,
$left\_each = left\_capa1 / left\_type$.

For $k' = k \sim k_{\max}$,

If $k' = k$, $d_{l,l',t}^{-k,k',r} += floor(left\_each)$, $left\_capa1 -= floor(left\_each)$.
Floor function represents the largest integer value which is smaller than $left\_each$.

Otherwise,

if $k' \neq k_{\max}$, $temp = ceil(left\_each)$.
if $(temp < left\_capa1)$, $d_{l,l',t}^{-k,k',r} = temp$; otherwise, $d_{l,l',t}^{-k,k',r} = left\_capa1$.

Otherwise, $d_{l,l',t}^{-k,k',r} = left\_capa1$.

$left\_capa1 -= d_{l,l',t}^{-k,k',r}$

End

End

Step 3: Assign $X_{l,l',t}^{k} = 0$ for all $k,l,l',t,l' \neq l$.

Step 4: For $t = 1 \sim t_{\max}$

For $k = 1 \sim k_{\max}$

For $l = 1 \sim l_{\max}$

$$I_{l,t}^{k} = \sum_{k'=1}^{k} \sum_{l'} \sum_{r} d_{l',l,(t-r)}^{-k',k,r} + I_{l,(t-1)}^{k} - \sum_{k'=1}^{k} \sum_{l'} \sum_{r} d_{l,l',t}^{-k',k,r} .$$

End

End

End

Step 5: For $t = 1 \sim t_{\max}$

For $k = 1 \sim k_{\max}$

For $l = 1 \sim l_{\max}$,

If $I_{l,t}^{k} < 0$, $\min\_h = -I_{l,t}^{k}$.

for $l' = 1 \sim l_{\max,}$ $l' \neq l$

for $t' = t \sim t_{\max}$

If $I_{l',t}^{k} \leq 0$, go back to forloop;

Otherwise, if $I_{l',t}^{k} < \min\_h$, let $\min\_h = I_{l',t}^{k}$.

End

$X_{l',l,t}^{k} = \min\_h$.

for $t' = t \sim t_{\max}$, $I_{l',t'}^{k} -= \min\_h, I_{l,t'}^{k} += \min\_h$

if $I_{l,t}^{k} < 0$, go back to forloop, otherwise, jump out from forloop.

End

End

End

End

Step 6: if $I_{l,t}^{k} < 0,\ \forall t,k,l$,

For $r = 1 \sim r_{\max}$

For $l' = 1 \sim l_{\max}$

For $k' = 1 \sim k_{\max}$

If capa1(k', r, t, l, l')>0, $\min\_h = \{capa1,\ d_{l,l',t}^{-k',k,r},\ (-I_{l,t}^{k})\}$. Then adjust other related I . Let $capa1 -= \min\_h$, $d_{l,l',t}^{-k',k,r} -= \min\_h$, $I_{l,t}^{k} += \min\_h$.

End

End

End

Step 7: if $I_{l,t}^{k} < 0,\ \forall t,k,l$,

For $k' = 1 \sim k$

For $r = 1 \sim r_{\max}$

If $d_{l,l,t}^{-k',k,r} > 0$, $\min\_h = \{d_{l,l,t}^{-k',k,r}, (-I_{l,t}^{k})\}$. Let $d_{l,l,t}^{-k',k,r} -= \min\_h$ and $I_{l,t}^{k} += \min\_h$. Then adjust other related I .

End

End

6.4.2 The Structure of Better Neighbors

There are three types of better neighbors in this problem based on the interchange of a single variable, $(d_{l,l',t}^{-k',k_1,r} - h, d_{l,l',t}^{-k',k_2,r} + h)$, $(X_{l,l',t}^{k} - h, I_{l,t}^{k} + h)$, and $(X_{l,l_1,t}^{k} - h, X_{l_2,l,t}^{k} - h)$. The better neighbors are obtained from the value of a specific variable adding/subtracting

a flexible value *r* and are only adopted for those which have better objectives than current solution. A flexible value *r* is decided based on the maximal reduction of the objective. The detailed structure is introduced below.

1) $(d_{l,l',t}^{-k',k_1,r} - h, d_{l,l',t}^{-k',k_2,r} + h)$

The prerequisite for this type of better neighbor is $d_{l,l',t}^{-k',k_1,r} > 0$, $k_1 > k_2$. $\min\_h = d_{l,l',t}^{-k',k_1,r}$. $diff_{obj} = v_{l,l',t}^{k_1,r} - v_{l,l',t}^{k_2,r}$

For $t' = t + r \sim t_{\max}$, if $I_{l',t'}^{k_1} > 0$, $diff_{obj} = diff_{obj} + m_{l',t'}^{k_1} - m_{l',t'}^{k_2}$

If $I_{l',t'}^{k_1} < \min\_h$, $\min\_h = I_{l',t'}^{k_1}$.

If $I_{l',t'}^{k_1} \le 0$, this neighbor is not selected.

For $t' = t \sim t_{\max}$, if $I_{l,t'}^{k_2} > 0$, $diff_{obj} = diff_{obj} + m_{l,t'}^{k_2} - m_{l,t'}^{k_1}$

If $I_{l,t'}^{k_2} < \min\_h$, $\min\_h = I_{l,t'}^{k_2}$.

If $I_{l,t'}^{k_2} \le 0$, this neighbor is not selected.

If $diff_{obj} > 0$, let $d_{l,l',t}^{-k',k_1,r} -= \min\_h$, For $t' = t + r \sim t_{\max}$, let $I_{l',t'}^{k_1} -= \min\_h$, . $I_{l',t'}^{k_2} += \min\_h$. The constraint representation of this case is

$$\sum_{k'=1}^{k_1}\sum_{l'}\sum_{r} d_{l',l,(t-r)}^{-k',k_1,r} + \sum_{l',l'\neq l} X_{l',l,t}^{k_1} + I_{l,(t-1)}^{k_1} = \sum_{k'=1}^{k_1}\sum_{l'}\sum_{r} d_{l,l',t}^{-k',k_1,r} \Downarrow + \sum_{l',l'\neq l} X_{l,l',t}^{k_1} + I_{l,t}^{k_1} \Uparrow \qquad (k_1, l, t)$$

$$\sum_{k'=1}^{k_1}\sum_{l'}\sum_{r} d_{l',l,(t-r+1)}^{-k',k_1,r} + \sum_{l',l'\neq l} X_{l',l,t+1}^{k_1} + I_{l,t}^{k_1} \Uparrow = \sum_{k'=1}^{k_1}\sum_{l'}\sum_{r} d_{l,l',(t+1)}^{-k',k_1,r} + \sum_{l',l'\neq l} X_{l,l',(t+1}^{k_1} + I_{l,(t+1)}^{k_1} \Uparrow \qquad (k_1, l, t+1)$$

$$\sum_{k'=1}^{k_1}\sum_{l}\sum_{r'} d_{l,l',(t+r-r')}^{-k',k_1,r'} \Downarrow + \sum_{l,l\neq l'} X_{l,l',(t+r)}^{k_1} + I_{l',(t+r-1)}^{k_1} = \sum_{k'=1}^{k_1}\sum_{l}\sum_{r'} d_{l',l,(t+r)}^{-k',k_1,r'} + \sum_{l,l\neq l'} X_{l',l,(t+r)}^{k_1} + I_{l',(t+r)}^{k_1} \Downarrow \qquad (k_1, l', t+r)$$

$$\sum_{k'=1}^{k_1}\sum_{l}\sum_{r'} d_{l,l',(t+r-r'+1)}^{-k',k_1,r'} + \sum_{l,l\neq l'} X_{l,l',(t+r+1)}^{k_1} + I_{l',(t+r)}^{k_1} \Downarrow = \sum_{k'=1}^{k_1}\sum_{l}\sum_{r'} d_{l',l,(t+r+1)}^{-k',k_1,r'} + \sum_{l,l\neq l'} X_{l',l,(t+r+1)}^{k_1} + I_{l',(t+r+1)}^{k_1} \Downarrow$$

$$(k_1, l', t+r+1)$$

$$\sum_{k'=1}^{k_2}\sum_{l'}\sum_{r} d_{l',l,(t-r)}^{-k',k_2,r} + \sum_{l',l'\neq l} X_{l',l,t}^{k_2} + I_{l,(t-1)}^{k_2} = \sum_{k'=1}^{k_2}\sum_{l'}\sum_{r} d_{l,l',t}^{-k',k_2,r} \Uparrow + \sum_{l',l'\neq l} X_{l,l',t}^{k_2} + I_{l,t}^{k_2} \Downarrow \qquad (k_2, l, t)$$

$$\sum_{k'=1}^{k_2}\sum_{l'}\sum_{r} d_{l',l,(t-r+1)}^{-k',k_2,r} + \sum_{l',l'\neq l} X_{l',l,t+1}^{k_2} + I_{l,t}^{k_2} \Downarrow = \sum_{k'=1}^{k_2}\sum_{l'}\sum_{r} d_{l,l',(t+1)}^{-k',k_2,r} + \sum_{l',l'\neq l} X_{l,l',(t+1}^{k_2} + I_{l,(t+1)}^{k_2} \Downarrow \qquad (k_2, l, t+1)$$

$$\sum_{k'=1}^{k_2}\sum_{l}\sum_{r'} d_{l,l',(t+r-r')}^{-k',k_2,r'} \Uparrow + \sum_{l,l\neq l'} X_{l,l',(t+r)}^{k_2} + I_{l',(t+r-1)}^{k_2} = \sum_{k'=1}^{k_2}\sum_{l}\sum_{r'} d_{l',l,(t+r)}^{-k',k_2,r'} + \sum_{l,l\neq l'} X_{l',l,(t+r)}^{k_2} + I_{l',(t+r)}^{k_2} \Uparrow \quad (k_2, l', t+r)$$

$$\sum_{k'=1}^{k_2}\sum_{l}\sum_{r'} d_{l,l',(t+r-r'+1)}^{-k',k_2,r'} + \sum_{l,l\neq l'} X_{l,l',(t+r+1)}^{k_2} + I_{l',(t+r)}^{k_2} \Uparrow = \sum_{k'=1}^{k_2}\sum_{l}\sum_{r'} d_{l',l,(t+r+1)}^{-k',k_2,r'} + \sum_{l,l\neq l'} X_{l',l,(t+r+1)}^{k_2} + I_{l',(t+r+1)}^{k_2} \Uparrow$$

$$(k_2, l', t+r+1)$$

2) $(X_{l,l',t}^{k} - h, I_{l,t}^{k} + h)$

The prerequisite for this type of better neighbor is $X_{l,l',t}^{k} > 0$, $l \neq l'$. $\min\_h = X_{l,l',t}^{k}$. $diff_{obj} = c_{l,l',t}^{k}$

For $t' = t \sim t_{\max}$, if $I_{l',t}^{k} > 0$, $diff_{obj} = diff_{obj} + m_{l',t'}^{k} - m_{l,t'}^{k}$.

If $I_{l',t'}^{k} < \min\_h$, let $\min\_h = I_{l',t'}^{k}$.

Otherwise, this neighbor is not selected.

If $diff_{obj} > 0$, let $X_{l,l',t}^{k} -= \min\_h$, For $t' = t \sim t_{\max}$, let $I_{l',t'}^{k} -= \min\_h$, $I_{l,t'}^{k} += \min\_h$. The constraint representation of this case is

$$\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d_{l',l,(t-r)}^{-k',k,r} + \sum_{l',l'\neq l} X_{l',l,t}^{k} + I_{l,(t-1)}^{k} = \sum_{k'=1}^{k}\sum_{l'}\sum_{r} d_{l,l',t}^{-k',k,r} + \sum_{l',l'\neq l} X_{l,l',t}^{k} \Downarrow + I_{l,t}^{k} \Uparrow \qquad (k, l, t)$$

$$\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d_{l',l,(t-r+1)}^{-k',k,r} + \sum_{l',l'\neq l} X_{l',l,t+1}^{k} + I_{l,t}^{k} \Uparrow = \sum_{k'=1}^{k}\sum_{l'}\sum_{r} d_{l,l',(t+1)}^{-k',k,r} + \sum_{l',l'\neq l} X_{l,l',(t+1)}^{k} + I_{l,(t+1)}^{k} \Uparrow \qquad (k, l, t+1)$$

$$\sum_{k'=1}^{k}\sum_{l}\sum_{r} d_{l,l',(t-r)}^{-k',k,r} + \sum_{l,l\neq l'} X_{l,l',t}^{k} \Downarrow + I_{l',(t-1)}^{k} = \sum_{k'=1}^{k}\sum_{l}\sum_{r} d_{l',l,t}^{-k',k,r} + \sum_{l,l\neq l'} X_{l',l,t}^{k} + I_{l',t}^{k} \Downarrow \qquad (k, l', t)$$

$$\sum_{k'=1}^{k}\sum_{l}\sum_{r} d_{l,l',(t-r+1)}^{-k',k,r} + \sum_{l,l\neq l'} X_{l,l',t+1}^{k} + I_{l',t}^{k} \Downarrow = \sum_{k'=1}^{k}\sum_{l}\sum_{r} d_{l',l,(t+1)}^{-k',k,r} + \sum_{l,l\neq l'} X_{l',l,(t+1)}^{k} + I_{l',(t+1)}^{k} \Downarrow \qquad (k, l', t+1)$$

3) $(X^k_{l,l_1,t}-h, X^k_{l_2,l,t}-h)$

The prerequisite for this type of better neighbor is $X^k_{l,l_1,t}>0$ and $X^k_{l_2,l,t}>0$. $\min\_h=\min\{X^k_{l,l_1,t}, X^k_{l_2,l,t}\}$. $diff_{obj}=c^k_{l,l_1,t}+c^k_{l_2,l,t}$.

For $t'=t\sim t_{\max}$, if $I^k_{l_1,t'}>0$, $diff_{obj}=diff_{obj}+m^k_{l_1,t'}-m^k_{l_2,t'}$.

If $I^k_{l_1,t'}<\min\_h$, let $\min\_h=I^k_{l_1,t'}$

Otherwise, this neighbor is not selected.

If $diff_{obj}>0$, let $X^k_{l,l_1,t}-=\min\_h$ and $X^k_{l_2,l,t}-=\min\_h$. For $t'=t\sim t_{\max}$, let $I^k_{l_1,t'}-=\min\_h,$. $I^k_{l_2,t'}+=\min\_h$. The constraint representation of this case is

$$\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l',l,(t-r)} + \sum_{l',l'\neq l} X^k_{l',l,t}[X^k_{l_2,l,t}\Downarrow] + I^k_{l,(t-1)} = \sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l,l',t} + \sum_{l',l'\neq l} X^k_{l,l',t}[X^k_{l,l_1,t}\Downarrow] + I^k_{l,t} \qquad (k,l,t)$$

$$\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l',l_1,(t-r)} + \sum_{l',l'\neq l} X^k_{l',l_1,t}\Downarrow + I^k_{l_1,(t-1)} = \sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l_1,l',t} + \sum_{l',l'\neq l} X^k_{l_1,l',t} + I^k_{l_1,t}\Downarrow \qquad (k,l_1,t)$$

$$\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l',l_2,(t-r+1)} + \sum_{l',l'\neq l} X^k_{l',l_2,t+1} + I^k_{l_2,t}\Downarrow = \sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l_2,l',(t+1)} + \sum_{l',l'\neq l} X^k_{l_2,l',(t+1)} + I^k_{l_2,(t+1)}\Downarrow \qquad (k,l_1,t+1)$$

$$\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l',l_2,(t-r)} + \sum_{l',l'\neq l} X^k_{l',l_2,t} + I^k_{l_2,(t-1)} = \sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l_2,l',t} + \sum_{l',l'\neq l} X^k_{l_2,l',t}\Downarrow + I^k_{l_2,t}\Uparrow \qquad (k,l_2,t)$$

$$\sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l',l_2,(t-r+1)} + \sum_{l',l'\neq l} X^k_{l',l_2,t+1} + I^k_{l_2,t}\Uparrow = \sum_{k'=1}^{k}\sum_{l'}\sum_{r} d^{-k',k,r}_{l_2,l',(t+1)} + \sum_{l',l'\neq l} X^k_{l_2,l',(t+1)} + I^k_{l_2,(t+1)}\Uparrow \qquad (k,l_2,t+1)$$

6.4.3 First-Improvement Descent Local Search

Based on the structure of better neighbors, a first-improvement descent local search is developed below.

Step 1: Let $Iter_{pre\_loop}=0$ and $Iter_{current}=0$. Go to Step 2.

Step 2:

For car type $k_1=2$ to k_{max}

For car type $k'=1$ to (k_1-1)

For rental period $r=1$ to r_{max}

For rental location $l=1$ to l_{max}

For rental location $l'=1$ to l_{max}

For time period $t=1$ to t_{max}

If any better neighbor i of $(d_{l,l',t}^{-k',k_1,r}-h,\ d_{l,l',t}^{-k',k_2,r}+h)$ is found,

move to i, $Iter_{current}= Iter_{current}+1$.

End

End

End

End

End

End

Step 3:

For car type $k=1$ to k_{max}

For rental location $l=1$ to l_{max}

For rental location $l' = 1$ to l_{max}

For time period $t = 1$ to t_{max}

If any better neighbor i of $(X^k_{l,l',t} - h,\ I^k_{l,t} + h)$ is found, move to i,

$Iter_{current} = Iter_{current} + 1$.

End

End

End

End

Step 4:

For car type $k = 1$ to k_{max}

For rental location $l = 1$ to l_{max}

For rental location $l' = 1$ to l_{max}

For time period $t = 1$ to t_{max}

If any better neighbor i of $(X^k_{l,l_1,t} - h,\ X^k_{l_2,l,t} - h)$ is found, move to i,

$Iter_{current} = Iter_{current} + 1$.

End

End

End

End

Step 5: If $Iter_{pre\_loop} \neq Iter_{current}$, let $Iter_{pre\_loop} = Iter_{current}$ and go to Step 2; otherwise, the whole algorithm is finished.

6.5 Computing Results

This section focuses on an experiment generated by two types of experimental factors including: the number of rental locations and (the number of time periods, the number of rental periods). Section 6.5.1 introduces the parameter setting and the meaning of the parameters. In section 6.5.2, the setting of factor levels and two parts of the experiment are described, including small scale problems and large scale problems. The software and the computer equipment used for conducting this experiment are described in Section 6.5.3 In Section 6.5.4, experimental results are analyzed and explained.

6.5.1 Parameter Settings

The parameter settings in demand allocation and empty flow redistribution are chosen based on the website data of Auto Rental News or the assumptions of making the problems feasible, and are as described below.

- **Demand**: The demand for car type 1 from node l to node l is generated from a uniform distribution (30, 50). The demand for car type 1 from node l to l' is generated from a uniform distribution (0, 20). As for the demands for car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (0.8, 1.1) for each car type. The demand for car type 2 is then generated by multiplying the demand for car type 1 by the ratio for car type 2. The demands for the higher car types are generated by multiplying demand for the previous car type by the respective ratio. The demand is rounded off to an integer.

- **Unit Inventory Cost:** The unit inventory cost of car type 1 is generated from a uniform distribution (5, 10). As for the unit inventory costs of car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (1.0, 1.2) for each car type. The unit inventory cost of car type 2 is then generated by multiplying the unit inventory cost of car type 1 by the ratio for car type 2. The unit inventory costs of the higher car types are generated by multiplying the unit inventory cost of the previous car type by the respective ratio. The unit inventory cost is rounded off to one decimal place.
- **Unit Operational Cost of A Rental Car:** The unit operational cost of a rental car of car type 1 from location l to location l' is generated by multiplying a random cost level, generated from an uniform distribution (30, 50), by rental period *r*. If location $l = l'$, a location ratio, generated from an uniform distribution (1.5, 3.0) is multiplied by the cost. As for the unit operational cost of a rental car of car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (1.2, 1.4) for each car type. The unit operational cost of a rental car of car type 2 is then generated by multiplying the unit rental car cost of car type 1 by the ratio for car type 2. The unit operational costs of a rental car of the higher car types are generated by multiplying unit operational cost of a rental car of the previous car type by the respective ratio. The unit operational cost of a rental car is rounded off to one decimal place.
- **Unit Operational Cost of An Empty Car:** The unit empty car cost of car type 1 is generated by multiplying a random distance between two locations, generated from the uniform distribution (0, 100), by a location ratio, generated from a uniform

distribution (0.3, 0.7). As for the unit empty car cost of car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (1.2, 1.4) for each car type. As for the unit rental car cost of car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (1.2, 1.4) for each car type. The unit empty car cost of car type 2 is then generated by multiplying the unit empty car cost of car type 1 by the ratio for car type 2. The unit empty car costs of the higher car types are generated by multiplying the unit empty car cost of the previous car type by the respective ratio. The unit empty car cost is rounded off to one decimal place.

- ✧ **Ratio of Satisfying Exact Demand without Upgrading:** In this problem, the ratio of satisfying exact demand without upgrading is set to be 0.5, which means at least 50% of the demand need to be satisfied without upgrading any car type. If this ratio is set to be higher, car rental companies will need to own more cars and many cars will be idle stored as the inventory. If this ratio is set to be lower, service level will be not important. Hence, we assume this ratio to be 50% as an average in this problem.
- ✧ **Initial Inventory**: The initial inventory of car type 1 is generated from a uniform distribution (0, 20). As for the initial inventory of car type 2 and the higher car types, a type ratio value is generated from a uniform distribution (0.9, 1.1) for each car type. The initial inventory of car type 2 is then generated by multiplying the initial inventory of car type 1 by the ratio for car type 2. The initial inventories of the higher car types are generated by multiplying the initial inventory of the previous car type by the respective ratio. The initial inventory is rounded off to an integer.

- **Initial Rental Car Flow:** The initial rental car flow $d\_pre^{-k_1,k_2,r}_{l_1,l_2,(t-r)\le 0}$ is generated based on the following rule. $ratio\_1$ and $ratio\_2$ are generated from the uniform distributions (0.8, 1.1) and (0.2, 0.5), individually. The initial rental car flow is rounded off to an integer.

If $l = l'$,

If $k_1 = 1$

If $k_2 = 1$, let $d\_pre^{-k_1,k_2,r}_{l_1,l_2,t}$ =uniform (30,50);

If $k_2 \neq 1$, $d\_pre^{-k_1,k_2,r}_{l_1,l_2,t} = ratio\_2 \times$ uniform (20,40).

Otherwise, if $k_1 = k_2$, $d\_pre^{-k_1,k_2,r}_{l_1,l_2,t} = ratio\_1 \times d\_pre^{-k_1,k_2-1,r}_{l_1,l_2,t}$.

If $k_1 \neq k_2$, $d\_pre^{-k_1,k_2,r}_{l_1,l_2,t} = ratio\_1 \times d\_pre^{-k_1-1,k_2,r}_{l_1,l_2,t}$

Otherwise, If $k_1 = 1$

If $k_2 = 1$, let $d\_pre^{-k_1,k_2,r}_{l_1,l_2,t}$ =uniform (0,20);

If $k_2 \neq 1$, $d\_pre^{-k_1,k_2,r}_{l_1,l_2,t} = ratio\_2 \times$ uniform (0,20).

Otherwise, if $k_1 = k_2$, $d\_pre^{-k_1,k_2,r}_{l_1,l_2,t} = ratio\_1 \times d\_pre^{-k_1-1,k_2-1,r}_{l_1,l_2,t}$.

If $k_1 \neq k_2$, $d\_pre^{-k_1,k_2,r}_{l_1,l_2,t} = ratio\_1 \times d\_pre^{-k_1-1,k_2,r}_{l_1,l_2,t}$

If $r \geq t$, let $d\_pre^{-k_1,k_2,r}_{l_1,l_2,(t-r)\leq 0} = d\_pre^{-k_1,k_2,r}_{l,l',(r-t)}$.

6.5.2 Factor Levels

Two parts of the experiment are exploited in demand allocation and empty flow redistribution. In part 1, the first-improvement descent local search is tested and compared to the branch-and-bound algorithm by the computing time and solution gap. In part 2, the algorithm time is tested in large scale problems in order to understand how large of a problem size this algorithm can solve.

Based on the experiment in part 1, the following 2 factors are tested. The number of car types is fixed at 10.

- ✧ **Number of Locations:** Three factor levels are set to be 6, 8, and 10, which are the maximal number of pools that can be solved by the branch-and-bound method through LINGO Software.

- ✧ **(Number of Time Periods, Number of Rental Periods):** Three factor levels are set to be (4, 4), (6, 6), and (8, 8).

In part 2, the large scale problems are tested to understand how large of a problem size this algorithm can solve and how long it will take. Normally a pool region includes 10~20 rental locations in a rental company and the planning periods is between 7~21. Hence, the problem size is assumed to be (number of nodes, number of time periods, number of rental periods). The problem sizes tested are (12,12,12),(15,15,15),(18,18,18), and (21,21,21).

6.5.3 Experimental Platform

This experiment uses the software Visual C++ to compile the computer coding of the best-improvement descent local search and uses the optimization software LINGO 9.0 to solve the optimal solution. The computer equipment for conducting this experiment includes an Intel Core 2 Duo E7400 2.80 GHz CPU and 6 GB memory.

6.5.4 Experimental Analysis

The experiment is divided into two parts. Small scale problems are generated to compare the computing times and the solution gaps between the first-improvement descent local search and the branch-and-bound method in part 1. Each factor contains three factor levels and each factor level uses three replications based on different random seeds. Hence, 3 factor levels of the number of locations × 3 factor levels of the number of time periods/rental periods × 3 random seeds = 27 independent trials in part 1.

In part 2, large scale problems are generated to determine how large of a problem size this algorithm can solve and how long it will take. Hence, 4 factor levels of the number of the problem sizes × 3 random seeds = 12 independent trials in part 2.

6.5.4.1 Results for Small Scale Problems

The experiment in part 1 was conducted for small scale problems. The solution gaps between the first-improvement descent local search and the branch-and-bound method are displayed in Table 6.1. The solution gap between the optimal solution and the initial solution is about 29%~30%. The first-improvement descent local search always obtained

extremely good solutions with an average solution gap below 0.07%. The computing times of the first-improvement descent local search and the branch-and-bound method are shown in Table 6.2. The branch-and-bound method solves fast in extremely small problems but takes significant amount of time when the problem size becomes larger. Instead, the first-improvement descent local search still solves quickly even when the problems become larger.

Table 6.1. Solution gaps between the first-improvement descent local search and the branch-and-bound method

Ex	# of Nodes	# of Time Periods / # ofRental Periods	Seed #	LINGO	First-Improvement Descent Local Search					
				Optimal Obj	Initial Solution			Final Solution		
					Initial Obj	Gap(%)	Avg Gap(%)	Final Obj	Gap(%)	Avg Gap(%)
1	6	4	20	3.2309E+07	4.1964E+07	29.88%	29.70%	3.2333E+07	0.07%	0.05%
2			40	3.4189E+07	4.4420E+07	29.92%		3.4214E+07	0.07%	
3			60	3.2525E+07	4.2048E+07	29.28%		3.2530E+07	0.01%	
4		6	20	1.0380E+08	1.3420E+08	29.28%	29.38%	1.0388E+08	0.07%	0.05%
5			40	1.0599E+08	1.3740E+08	29.63%		1.0601E+08	0.01%	
6			60	1.0077E+08	1.3023E+08	29.24%		1.0082E+08	0.05%	
7		8	20	2.3093E+08	2.9955E+08	29.72%	29.75%	2.3096E+08	0.02%	0.03%
8			40	2.3814E+08	3.0927E+08	29.87%		2.3821E+08	0.03%	
9			60	2.3372E+08	3.0308E+08	29.68%		2.3383E+08	0.05%	
10	8	4	20	5.3724E+07	6.9344E+07	29.08%	29.30%	5.3752E+07	0.05%	0.04%
11			40	5.6239E+07	7.3007E+07	29.82%		5.6264E+07	0.04%	
12			60	5.3823E+07	6.9434E+07	29.00%		5.3836E+07	0.02%	
13		6	20	1.7110E+08	2.2202E+08	29.76%	29.69%	1.7113E+08	0.02%	0.03%
14			40	1.7634E+08	2.2903E+08	29.88%		1.7644E+08	0.06%	
15			60	1.6923E+08	2.1902E+08	29.42%		1.6926E+08	0.02%	
16		8	20	3.9696E+08	5.1580E+08	29.94%	29.91%	3.9705E+08	0.02%	0.03%
17			40	4.0018E+08	5.1908E+08	29.71%		4.0041E+08	0.06%	
18			60	3.9606E+08	5.1523E+08	30.09%		3.9613E+08	0.02%	
19	10	4	20	8.4138E+07	1.0909E+08	29.65%	29.73%	8.4162E+07	0.03%	0.03%
20			40	8.6324E+07	1.1219E+08	29.96%		8.6360E+07	0.04%	
21			60	8.2530E+07	1.0693E+08	29.56%		8.2545E+07	0.02%	
22		6	20	2.6045E+08	3.3823E+08	29.86%	29.83%	2.6052E+08	0.03%	0.03%
23			40	2.6732E+08	3.4678E+08	29.73%		2.6743E+08	0.04%	
24			60	2.6203E+08	3.4041E+08	29.91%		2.6211E+08	0.03%	
25		8	20	6.0258E+08	7.8454E+08	30.20%	29.97%	6.0354E+08	0.16%	0.07%
26			40	6.0613E+08	7.8695E+08	29.83%		6.0628E+08	0.02%	
27			60	6.0592E+08	7.8706E+08	29.89%		6.0611E+08	0.03%	

Table 6.2. Computing times of the first-improvement descent local search and the branch-and-bound method

Ex	# of Nodes	# of Time Periods/ # of Rental Periods	Seed #	# of Variables	# of Constraints	LINGO		First-Improvement Descent Local Search			
						Run time(s)	Avg Time(s)	Iterations	Avg Iterations	Run time(s)	Avg Time(s)
1	6	4	20	33120	6011	4	4.0	14293	14723	5.7	5.7
2			40			4		15240		5.9	
3			60			4		14637		5.6	
4		6	20	73440	13331	22	20.3	32284	32895	8.7	8.8
5			40			19		33886		9.2	
6			60			20		32515		8.7	
7		8	20	129600	23531	78	90.7	57436	58502	13.3	13.5
8			40			95		60069		13.8	
9			60			99		58002		13.5	
10	8	4	20	58880	10571	10	11.0	24124	24942	7.3	7.9
11			40			12		25800		7.5	
12			60			11		24902		9.1	
13		6	20	130560	23531	80	77.7	55815	56398	15.8	13.9
14			40			85		58017		13.2	
15			60			68		55361		12.8	
16		8	20	230400	41611	511	581.3	99931	101047	20.2	21.4
17			40			619		102350		22.3	
18			60			614		100861		21.8	
19	10	4	20	92000	16411	30	28.0	37964	38765	10.3	10.4
20			40			28		40087		10.9	
21			60			26		38244		10.2	
22		6	20	204000	36611	179	211.3	85277	86557	19.5	18.9
23			40			232		88235		18.8	
24			60			223		86160		18.5	
25		8	20	360000	64811	1327	1309.0	154083	155619	33.0	34.2
26			40			1393		156790		39.5	
27			60			1207		155985		30.0	

6.5.4.2 Results for Large Scale Problems

The experiment in part 2 is conducted for large scale problems. Table 6.3 represents the algorithm time on large problem sizes. The largest cases, (21, 21 , 21), covers 10.79 million integer variables and 1.95 million constraints and the first-improvement descent local search only takes about 23 minutes to solve. The trend chart of algorithm time and the number of iterations for large problem sizes is presented in Figure 6.2.

Table 6.3. The algorithm time on large problem sizes

Ex	(Nodes, Time Periods, Rental Periods)	Seed #	# of Variables (unit: million)	# of Constraints (unit: million)	First-Improvement Descent Local Search			
					# of Iterations	Avg Iterations	Run time(s)	Avg Time(s)
1	(12, 12, 12)	20	1.16	0.21	497	500	135	138
2		40			504		144	
3		60			499		134	
4	(15, 15, 15)	20	2.82	0.51	1207	1208	412	384
5		40			1215		361	
6		60			1203		377	
7	(18, 18, 18)	20	5.83	1.05	2476	2483	979	926
8		40			2493		898	
9		60			2479		901	
10	(21, 21, 21)	20	10.79	1.95	4562	4567	1355	1353
11		40			4579		1351	
12		60			4560		1355	

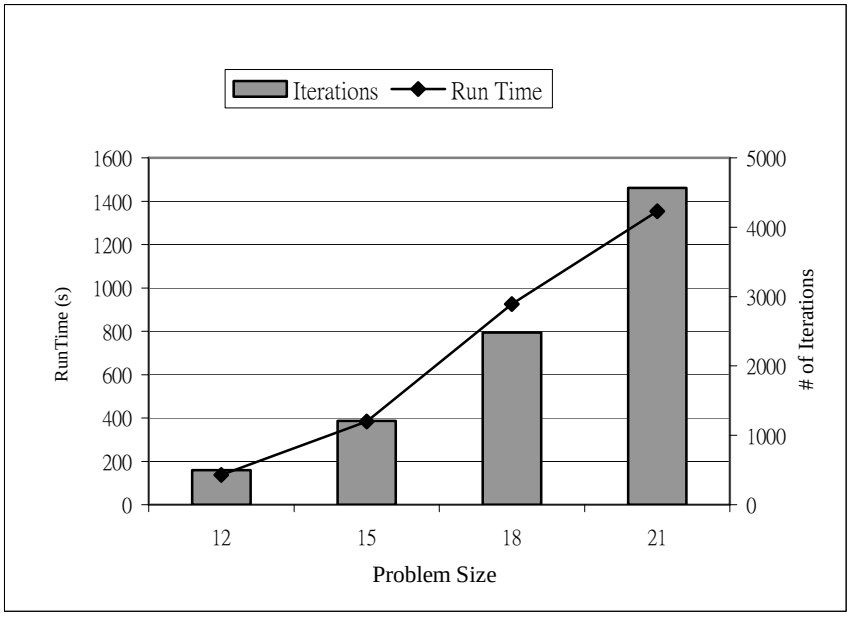

Figure 6.2. Computing time and iterations on large problem sizes

6.6 Concluding Remarks

In this chapter, a model for daily planning of demand allocation and empty flow redistribution was proposed. Empty cars were assumed to be redistributed the same day in the same pool. Car upgrade policy and service level were considered. A first-improvement descent local search with the structure of three interchange better neighbors was introduced and validated.

Based on the numerical results, in small scale problems, the first-improvement descent local search always obtains good solutions with an extremely small solution gap below 0.07% and takes significantly less time than the branch-and-bound method as the problem size increases. In large scale problems, the first-improvement descent local search only needs 23 minutes to solve the largest cases, (21,21,21), which covers 10.79 million integer variables and 1.95 million constraints. These results show that the first-improvement descent local search not only obtains a relatively good solution in a quite short time, but also solves very large scale integer programming problems easily.

CHAPTER 7

SUMMARY AND FUTURE RESEARCH

7.1 Summary

In this dissertation, the background knowledge of the car rental business was briefly introduced and the entire outlook of fleet planning in the car rental business was outlined. The car rental business profile in the United States was provided by discussing the history of the development of the car rental business, the statistics of the U.S. car rental market, and the functions of car rental software. Relevant problems of car rentals and fleet planning have been presented as well as part of an in-depth literature review.

A thorough analytical framework for car rental fleet planning in different time phases was built. In order to effectively solve different time phase problems in actual problem sizes, three practical algorithm procedures were developed.

In long-term planning, a binary integer model of pool segmentation and hub selection was formulated similarly to the capacitated facility location model with a single source constraint. All locations are split into several regions and one location is selected from each region to be the regional hub center. A clustering-based iterative algorithm, utilizing three important modules, was proposed and validated. The clustering algorithm uses the concept of unit demand cost to cluster nearby locations and quickly captures a very good initial solution. The iterative procedure of an enumeration method and a

modified Prim's algorithm utilizes the concept of a convex function to achieve a near-optimal solution. Numerical results show that the branch-and-bound method takes 100~1000 times the algorithm time to find a better solution and 100~10,000 times the algorithm time to find an optimal solution than in the clustering-based iterative algorithm. In addition, the solution gap of the clustering-based iterative algorithm is relatively small. The largest case tested involves 6000 nodes and 36 million integer variables, which is close to the level of Enterprise Rent-A-Car, the top car rental company in the United States. The clustering-based iterative algorithm takes about 135 minutes to solve a problem of this size. Based on the numerical results, it is clear that the clustering-based iterative algorithm not only can obtain satisfactory solutions with small solution gaps rapidly, but also readily solves very large scale problems.

In mid-term planning, inter-pool moves are considered, and buying/selling cars are allocated to different pool regions based on the change of seasonal demand. An integer programming model is developed and a best-improvement descent local search with the structure of better neighbors was introduced. The structure of better neighbors exploits the concept of maximal objective reduction to determine a flexible value h in the neighbors $Var \pm h$. In addition, better neighbors exploit the inventory balance constraints to adjust other variables and largely reduce the computing time of evaluating the constraints. The numerical results show that the solution gap normally falls below 4%. Based on the computational results, if the problem size becomes larger, the solution gap becomes smaller. The average time required by the branch-and-bound method is about 50 to 380 times more than the time of the best-improvement descent local search. The largest problem size tested contains 200 pools, 46 million integer variables, and 250

thousand constraints, and the best-improvement descent local search only takes 38.6 seconds to solve. These results demonstrate that the best-improvement descent local search has very good performance and can obtain a near-optimal solution in an extremely short time.

In short-term planning, daily planning of demand allocation and empty flow redistribution was addressed in the same pool region. The fact that the customers can rent a car and return it either in the same or in a different location leads to the problem of vehicle imbalance, and empty vehicles need to be redistributed. In the same pool, empty cars can be redistributed the same day. Car upgrade policy and service level are also considered. This problem is formulated as an integer programming model and a first-improvement descent local search with the structure of three interchange better neighbors is presented. The numerical results show that the first-improvement descent local search always obtains a good solution quickly with an exceedingly small solution gap below 0.07%. The first-improvement descent local search only takes 23 minutes to solve the largest case, (21,21,21), which covers 10.79 million integer variables and 1.95 million constraints. These results demonstrate that the first-improvement descent local search not only obtains a relatively good solution in a quite short time but also solves very large scale integer programming problems easily.

7.2 Directions for Further Extensions and Research

In this dissertation, a set of deterministic car rental fleet plans was developed. However, such a set of models is a basic framework. Still other parts need to be considered, such as demand patterns, unmet demand, random customers (i.e. walk-in customers without prior reservations), cost and pricing structure, demand allocation and empty flow redistribution at different pool regions.

In the design of the algorithms discussed in this dissertation, the clustering algorithm is a simple but powerful method of the initial solution. A good feasible solution can be obtained in an extremely short time. If the clustering algorithm and an impactful heuristic developed can effectively be applied to a general capacitated facility location problem, it may solve very large scale optimization problems easily. This promises to be a very viable research direction.

The structure of better neighbors can be utilized on the problem with flow balance constraints. In the mid-term and short-term fleet planning, the structure of better neighbors has been used in the design of the best-improvement descent local search and the first-improvement descent local search. The structure of better neighbors is expected to employ on more problems with flow balance constraints and a more complete and more general structure of better neighbors can then be integrated.

REFERENCES

Ahuja, R.K., Orlin, J.B., Pallottino, S., Scaparra, M.P., and Scutella, M.G., 2004. A multi-exchange heuristic for the single-source capacitated facility location problem. *Management Science*, 50(6), 749-760.

Anderson, C.K., Davison, M., and Rasmussen, H., 2004. Revenue management: a real options approach. *Naval Research Logistics*, 51(5), 686-703.

Aykin, T., 1995. Networking policies for hub-and-spoke systems with application to the air transportation system. *Transportation Science*, 29(3), 201-221.

Babkin, V.T., Gasretov, A.L., and Zolotukhin, V.F., 1977. Branch-and-bound algorithm for solving the generalized problem of optimal assignment. *Engineering Cybemetics*, 15(6), 37-42.

Barnett, A., 2000. Free-flight and en route air safety: A first-order analysis. *Operations Research*, 48(6), 833-845.

Barnhart, C., Belobaba, P., and Odoni, A.R., 2003. Applications of operations research in the air transport industry. *Transportation Science*, 37(4), 368-391.

Baxley, B.T., Williams, D., Consigilio, M., Adams, C., and Abbott, T., 2008. Small aircraft transportation system, higher volume operations concept and research summary. *Journal of Aircraft*, 45(6), 1825-1834.

Beaujon, G.J., and Turnquist, M.A., 1991. A model for fleet sizing and vehicle allocation. *Transportation Science*, 25(1), 19-45.

Bertsimas, D., and Popescu, I., 2003. Revenue management in a dynamic network environment. *Transportation Science*, 37(3), 257-277.

Bojovic, N.J., 2002. A general system theory approach to rail freight car fleet sizing. *European Journal of Operational Research*, 136(1), 136-172.

Cao, J.M., and Kanafani, A., 2000. Value of runway time slots for airlines. *European Journal of Operational Research*, 126(3), 491-500.

Carroll, W.J., and Grimes, R.C., 1995. Evolutionary change in product management: experiences in the car rental industry. *Interfaces*, 25(5), 84-104.

Caseau, Y., and Kokeny, T., 1998. An inventory management problem. *Constraints*, 3(4), 363-373.

Chatwin, R.E., 1996. Multi-period airline overbooking with multiple fare classes. *Naval Research Logistics*, 43(5), 603-612.

Chatwin, R.E., 1998. Multi-period airline overbooking with a single fare class. *Operations Research*, 46(6), 805-819.

Chen, C.H., and Ting, C.J., 2006. Applying multiple ant colony system to solve single source capacitated facility location problem. *Ant Colony Optimization and Swarm Intelligence, 5th International Worksho*p, *ANTS 2006, Brussels, Belgium, September 4-7, 2006 Proceedings*, 4150, 508-509.

Cho, S., and Rust, J., 2008. Is econometrics useful for private policy making? A case study of replacement policy at an auto rental company. *Journal of Econometrics*, 145(1-2), 243-257.

Cooper, W.M., 2002. Asymptotic behavior of an allocation policy for revenue management. *Operations Research*, 50(4), 720-727.

Correia, I., and Captivo, M.E., 2006. Bounds for the single source modular capacitated plant location problem. *Computers & Operations Research*, 33(10), 2991-3003.

Couillard, J., and Martel, A., 1990. Vehicle fleet planning in the road transportation industry. *IEEE Transactions on Engineering Management*, 37(1), 31-36.

Dejax, P.J., and Crainic, T.G., 1987. A review of empty flows and fleet management models in freight transportation. *Transportation Science*, 21(4), 227-247.

Dodgson, J.S., 1994. Competition policy and the liberalization of European aviation. *Transportation*, 21(4), 355-370.

Donohue, G.L., 2006. U.S. air transportation: An approaching perfect storm. *Aerospace America*, 44(8), 26-32.

Dorigo, M., Maniezzo, and V., Colorni, A., 1996. The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on System, Man, and Cybernetics*, 26(1), 29-41.

Doy, G., and Pope, J., 1985. Regional development and regional air transport in the European economic community. *Revue T, Revue Belge du Transport*, 2, 181-185.

Du, Y., and Hall, R., 1997. Fleet sizing and empty equipment redistribution for center-terminal transportation networks. *Management Science*, 43(2), 145-157.

Edelstein, M., and Melnyk, M., 1977. The pool control system. *Interfaces*, 8(1), 21-36.

Elhedhli, S., and Goffin, J.L., 2004. The integration of an interior-point cutting plane method with a branch-and-price algorithm. *Mathematical Programming*, 100(2), 267-294.

Fei, H., Chu, C., Meskens, N., and Artiba, A., 2008. Solving surgical cases assignment problem by a branch-and-price approach. *International Journal of Production Economics*, 112(1), 96-108.

Feo, T., and Resende, M., 1995. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6, 109-133.

Fink, A., and Reiners, T., 2006. Modeling and solving the short-term car-rental logistics problem. *Transportation Research Part E: Logistics and Transportation Review*, 42(4), 272-292.

Furth, P.G., and Nash, A.B., 1985. Vehicle pooling in transit operations. *Journal of Transportation Engineering*. 111(3), 268-279.

Geraghty, M.K., and Johnson, E., 1997. Revenue management saves National Car Rental. *Interfaces*, 27(1), 107-127.

Glover, F., 1977. Heuristics for integer programming using surrogate constraints. *Decision Sciences*, 8(1), 156-166.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. *Computer & Operations Research*, 13, 533-549.

Glover, F., 1998. A template for scatter search and path relinking. *Lecture Notes in Computer Science*, 1363, 13.

Goodovitch, T., 1996. Theory of air transport development. *Transportation Planning and Technology*, 20(1), 1-13.

Haddadi, S., and Ouzia, H., 2004. Effective algorithm and heuristic for the generalized assignment problem. *European Journal of Operational Research*, 153(1), 184-190.

Hall, R.W., 1999. Stochastic freight flow patterns: implications for fleet optimization. *Transportation Research Part A: Policy and Practice*, 33(6), 449-465.

Holland, J., 1975. Adaptation in natural and artificial systems. *Ann Arbor: University of Michigan Press.*

Holmberg, K., Ronnqvist, M., and Yuan, D., 1999. An exact algorithm for the capacitated facility location problems with single sourcing. *European Journal of Operational Research*, 113(3), 544-559.

Hong, W.C., Lai, Y.J., Pai, P.F., Lee, S.L., and Yang, S.L., 2007. Composite of support vector regression and evolutionary algorithms in car-rental revenue forecasting. *2007 IEEE Congress on Evolutionary Computation*, 2872-2878.

Hsu, C.I., and Wen, Y.H., 2000. Application of Grey theory and multiobjective programming towards airline network design. *European Journal of Operational Research*, 127(1), 44-68.

Inaba, T., 2008. Value of sparse RFID traceability information in asset tracking during migration period. *2008 IEEE International Conference on RFID*, 183-190.

Janic, M., 2003. Modelling operational, economic and environmental performance of an air transport network. *Transportation Research Part D: Transport and Environment*, 8(6), 415-432.

Jarrah, A.I.Z., Yu, G., Krishamurthy, N., and Rakshit, A., 1993. Decision support framework for airline flight cancellations and delays. *Transportation Science*, 27(3), 266-280.

Jenkins, L., 1987. Using parametric integer programming to plan the mix of an air transport fleet. *INFOR: Information Systems and Operational Research*, 25(2), 117-135.

Joborn, M., Crainic, T.G., Gendreau, M., Holmberg, K., and Lundgren, J.T., 2004. Economies of scale in empty freight car distribution in scheduled railways. *Transportation Science*, 38(2), 121-134.

Karaesmen, I., and Ryzin, G., 2004. Overbooking with substitutable inventory classes. *Operations Research*, 52(1), 83-104.

Kelley, J.E. Jr., 1960. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4), 703-712.

Khoury, H.M., Kamat, V.R., and Ioannou, P.G., 2007. Evaluation of general-purpose construction simulation and visualization tools for modeling and animating airside airport operations. *Simulation*, 83(9), 663-679.

Kirkpatrick, S., Gelatt, C., and Vecchi, M., 1983. Optimization by simulated annealing. *Science*, 220, 671-680.

Klose, A., 1999. An LP-based heuristic for two-stage capacitated facility location problems. *Journal of the Operational Research Society*, 50(2), 157-166.

Kochel, P., Kunze, S., and Nielander, U., 2003. Optimal control of a distributed service system with moving resources: application to the fleet sizing and allocation problem. *International Journal of Production Economics*, 81-82, 443-459.

Kuyumcu, A., and Garcia-Diaz, A., 2000. A polyhedral graph theory approach to revenue management in the airline industry. *Computers & Industrial Engineering*, 38(3), 375-396.

Li, Y., and Wang, X., 2005. Integration of fleet assignment and aircraft routing. *Transportation Research Record*, 1915, 79-84.

Lines, L., Kuby, M., Schultz, R., Clancy, J., and Xie, Z., 2008. A rental car strategy for commercialization of hydrogen in Florida. *International Journal of Hydrogen Energy*, 33(20), 5312-5325.

Lohatepanont, M., and Barnhart, C., 2004. Airline schedule planning: Integrated models and algorithms for schedule design and fleet assignment. *Transportation Science*, 38(1), 19-32.

Lourenco, H.R., Martin, O.C., Stutzle, T., 2002. Iterated local search. Glover F., and Kochenberger, G., editors, *Handbook of Metaheuristics,* 321-353. Kluwer Academic Publishers.

Marker, J.L., 1991. Flight operations safety management. *SAE Transactions,* 100(1), 2433-2442.

Martel, A., 1990. Vehicle fleet planning in the road transportation industry. *IEEE Transactions on Engineering Management,* 37(1), 31-36.

Mladenovic, N., and Hansen, P., 1997. Variable neighborhood search. *Computers & Operations Research*, 24(11), 1097-1100.

Mladenovic, N., Brimberg, J., Hansen, P., and Moreno-Perez, J.A., 2007. The p-median problem: a survey of metaheuristic approaches. *European Journal of Operational Research*, 179(3), 927-939.

Moscato, P., 1989. On evolution search optimization, genetic algorithms and martial arts: towards memetic algorithms. *Technical Report Caltech Concurrent Computation Program, Report. 826. Pasadena, CA: California Institute of Technology.*

Nauss, R.M., 2004. The elastic generalized assignment problem. *Journal of the Operational Research Society*, 55(12), 1333-1341.

Nero, G., and Black, J.A., 1998. Hub-and-spoke networks and the inclusion of environmental costs on airport pricing. *Transportation Research Part D: Transport and Environment,* 3(5), 275-296.

Netessine, S., Dobson, G., and Shumsky, R.A., 2002. Flexible service capacity: optimal investment and the impact of demand correlation. *Operations Research*, 50(2), 375-388.

New, S., 2003. Multimedia for international operations: a case study. *International Journal of Operations & Production Management*, 23(1), 125-137.

Orlady, H.W., and Orlady, L.M., 2002. Human factors in multi-crew flight operations. *Aeronautical Journal*, 106(1060), 321-324.

Osman, I., and Ahmadi, S., 2007. Guided construction search metaheuristics for the capacitated p-median problem with single source constraint. *Journal of the Operational Research Society*, 58(1), 100-114.

Pachon, J.E., 2000. Strategic and tactical fleet planning for the car rental industry. *Ph.D. Dissertation, University of Miami, Coral Gables, Florida.*

Pachon, J.E., Iakovou, E., IP, C., and Aboudi, R., 2003. A synthesis of tactical fleet planning models for the car rental industry. *IIE Transactions*, 35(9), 907-916.

Parikh, S.C., 1977. On a fleet sizing and allocation problem. *Management Science*, 23(9), 972-977.

Park, J.S., Lim, B.H., and Lee, Y., 1998. Lagrangian dual-based branch-and-bound algorithm for the generalized multi-assignment problem. *Management Science*, 44(12), 271-282.

Pentico, D.W., 2007. Assignment problems: a golden anniversary survey. *European Journal of Operational Research*, 176(2), 774-793.

Pirkul, H., 1987. Efficient algorithm for the capacitated concentrator location problem. *Computers & Operations Research*, 14(3), 197-208.

Ronnqvist, M., Tragantalerngsak, S., and Holt, J., 1999. A repeated matching heuristic for the single-source capacitated facility location problem. *European Journal of Operational Research*, 116(1), 51-68.

Rosenthal, R.E., and Waslsh, W.J., 1996. Optimizing flight operations for an aircraft carrier in transit. *Operations Research*, 44(2), 305-312.

Sambracos, E., Paravantis, J.A., Tarantilis, C.D., and Kiranoudis, C.T., 2004. Dispatching of small containers via coastal freight liners: the case of the Aegean Sea. *European Journal of Operational Research*, 152(2), 365-381.

Shi, L., Olafsson, S., 2000. Nested partitions method for global optimization. *Operations Research*, 48(3), 390-407.

Song, D.P., and Earl, C.F., 2008. Optimal empty vehicle repositioning and fleet-sizing for two-depot service systems. *European Journal of Operational Research*, 185(2), 760-777.

Sridharan, R., 1993. A Lagrangian heuristic for the capacitated plant location problem with single source constraints. *European Journal of Operational Research*, 66(3), 305-312.

Stickle, J.W., Stewart, R.J., and Holmes, B.J., 1991. *Aerospace America*, 29(9), 56-58.

Subramanian, J., Stidham Jr., S., and Lautenbacher, C.J., 1999. Airline yield management with overbooking, cancellations, and no-shows. *Transportation Science*, 33(2), 147-167.

Syam, S.S., 1997. Model for the capacitated p-facility location problem in global environments. *Computers & Operations Research*, 24(11), 1005-1016.

Tainiter, M., 1964. Some stochastic inventory models for rental situations. *Management Science*, 11(2), 316-326.

Tyler, J.E. Jr., 1986. Maximizing the physical capacity of a fleet of aircraft and containers handling mixed freight. *SAE Technical Paper Series.*

Voudouris, C., and Tsang, E., 1996. Partial constraint problems and guided local search. *Proceeding of the Second International Conference on the Practical Application of Constraint Technology*, 337-356.

Wu, G.H., 2009. Fleet planning in the car rental business. *Working Ph.D. Dissertation, The Pennsylvania State University, State College, Pennsylvania.*

Wu, P., Hartman, J.C., and Wilson, G.R., 2003. A demand-shifting feasibility algorithm for Benders decomposition. *European Journal of Operational Research*, 148(3), 570-583.

Wu, P., Hartman, J.C., and Wilson, G.R., 2005. An integrated model and solution approach for fleet sizing with heterogeneous assets. *Transportation Science*, 39(1), 87-103.

Yan, S., and Young, H.F., 1996. Decision support framework for multi-fleet routing and multi-stop flight scheduling. *Transportation Research Part A: Policy and Practice*, 30(5), 379-398.

Yan, S., and Tu, Y.P., 1997. Multifleet routing and multistop flight scheduling for schedule perturbation. *European Journal of Operational Research*, 103(1), 155-169.

APPENDIX A

Original Data on Experimental Factors versus Algorithm Time in Chapter 4

EX	# of Types	# of Locations	Seed #	R_Hubcost	R_Cap	Obj	Time
1	4	300	20	300	10	1,275,865.630	1.713
2	4	300	20	300	25	852,556.630	1.507
3	4	300	20	300	62.5	834,005.310	1.786
4	4	300	20	600	10	2,133,898.750	3.408
5	4	300	20	600	25	1,173,806.880	1.726
6	4	300	20	600	62.5	1,097,309.250	3.142
7	4	300	20	1200	10	3,797,481.750	5.460
8	4	300	20	1200	25	1,737,567.880	1.928
9	4	300	20	1200	62.5	1,432,009.750	4.636
10	4	300	40	300	10	1,275,151.380	1.686
11	4	300	40	300	25	889,750.310	1.584
12	4	300	40	300	62.5	880,294.130	1.981
13	4	300	40	600	10	2,117,378.500	2.697
14	4	300	40	600	25	1,220,462.000	1.745
15	4	300	40	600	62.5	1,169,976.130	3.305
16	4	300	40	1200	10	3,709,916.250	5.522
17	4	300	40	1200	25	1,823,104.500	2.142
18	4	300	40	1200	62.5	1,519,524.500	4.825
19	4	300	60	300	10	1,126,611.380	1.384
20	4	300	60	300	25	816,238.060	1.443
21	4	300	60	300	62.5	804,893.940	1.816
22	4	300	60	600	10	1,797,369.250	2.177
23	4	300	60	600	25	1,115,333.250	1.598
24	4	300	60	600	62.5	1,060,959.380	2.839
25	4	300	60	1200	10	3,177,877.750	4.865
26	4	300	60	1200	25	1,652,163.380	1.871
27	4	300	60	1200	62.5	1,384,964.880	4.343
28	4	600	20	300	10	2,172,480.250	15.276
29	4	600	20	300	25	1,387,468.250	10.990
30	4	600	20	300	62.5	1,346,308.750	16.054
31	4	600	20	600	10	3,773,232.000	36.243
32	4	600	20	600	25	1,966,396.380	10.885
33	4	600	20	600	62.5	1,717,856.500	22.500
34	4	600	20	1200	10	6,935,220.500	64.000
35	4	600	20	1200	25	2,998,268.250	11.930
36	4	600	20	1200	62.5	2,317,366.750	23.623
37	4	600	40	300	10	2,076,626.500	10.122
38	4	600	40	300	25	1,384,667.880	10.246
39	4	600	40	300	62.5	1,378,016.630	15.173
40	4	600	40	600	10	3,524,143.250	28.207
41	4	600	40	600	25	1,939,269.500	10.994
42	4	600	40	600	62.5	1,788,635.000	21.939
43	4	600	40	1200	10	6,460,248.000	53.785
44	4	600	40	1200	25	2,919,466.500	14.622
45	4	600	40	1200	62.5	2,265,393.000	25.986
46	4	600	60	300	10	2,458,591.250	23.408
47	4	600	60	300	25	1,413,733.630	10.583
48	4	600	60	300	62.5	1,317,975.880	16.138
49	4	600	60	600	10	4,335,933.500	47.505
50	4	600	60	600	25	2,081,489.380	10.389

EX	# of Types	# of Locations	Seed #	R_Hubcost	R_Cap	Obj	Time
51	4	600	60	600	62.5	1,720,150.880	20.353
52	4	600	60	1200	10	8,169,460.000	88.887
53	4	600	60	1200	25	3,274,164.750	18.767
54	4	600	60	1200	62.5	2,270,872.250	19.893
55	4	1200	20	300	10	3,932,258.000	189.136
56	4	1200	20	300	25	2,281,679.500	71.539
57	4	1200	20	300	62.5	2,179,808.000	122.751
58	4	1200	20	600	10	7,016,587.500	549.429
59	4	1200	20	600	25	3,285,557.250	85.235
60	4	1200	20	600	62.5	2,798,644.000	163.030
61	4	1200	20	1200	10	12,923,384.000	1,015.197
62	4	1200	20	1200	25	5,136,709.000	139.627
63	4	1200	20	1200	62.5	3,647,079.750	168.660
64	4	1200	40	300	10	4,345,223.500	250.551
65	4	1200	40	300	25	2,403,136.250	74.700
66	4	1200	40	300	62.5	2,234,461.500	121.361
67	4	1200	40	600	10	7,777,016.000	597.769
68	4	1200	40	600	25	3,549,647.000	85.036
69	4	1200	40	600	62.5	2,929,204.750	165.108
70	4	1200	40	1200	10	14,662,852.000	1,089.421
71	4	1200	40	1200	25	5,569,483.500	162.416
72	4	1200	40	1200	62.5	3,825,088.500	153.360
73	4	1200	60	300	10	4,089,390.250	204.641
74	4	1200	60	300	25	2,435,587.500	80.682
75	4	1200	60	300	62.5	2,303,455.250	135.860
76	4	1200	60	600	10	7,232,989.500	460.928
77	4	1200	60	600	25	3,536,960.750	102.001
78	4	1200	60	600	62.5	2,873,689.000	157.488
79	4	1200	60	1200	10	13,500,923.000	854.174
80	4	1200	60	1200	25	5,517,274.000	161.292
81	4	1200	60	1200	62.5	3,708,450.500	157.422
82	8	300	20	300	10	1,412,112.380	1.598
83	8	300	20	300	25	1,077,365.500	1.324
84	8	300	20	300	62.5	1,070,129.500	1.565
85	8	300	20	600	10	2,306,782.000	2.510
86	8	300	20	600	25	1,503,237.130	1.667
87	8	300	20	600	62.5	1,437,789.130	2.310
88	8	300	20	1200	10	4,028,794.250	4.364
89	8	300	20	1200	25	2,094,002.130	1.738
90	8	300	20	1200	62.5	1,867,862.130	3.462
91	8	300	40	300	10	1,236,389.130	1.096
92	8	300	40	300	25	1,067,265.000	1.435
93	8	300	40	300	62.5	1,066,851.500	1.667
94	8	300	40	600	10	1,896,165.130	1.661
95	8	300	40	600	25	1,444,201.750	1.980
96	8	300	40	600	62.5	1,426,330.750	2.349
97	8	300	40	1200	10	3,161,869.500	2.518
98	8	300	40	1200	25	2,000,696.630	2.162
99	8	300	40	1200	62.5	1,848,918.130	4.147
100	8	300	60	300	10	1,145,600.000	1.262

EX	# of Types	# of Locations	Seed #	R_Hubcost	R_Cap	Obj	Time
101	8	300	60	300	25	988,155.940	1.601
102	8	300	60	300	62.5	979,013.250	1.765
103	8	300	60	600	10	1,765,742.880	1.601
104	8	300	60	600	25	1,346,354.250	2.062
105	8	300	60	600	62.5	1,317,505.250	2.467
106	8	300	60	1200	10	2,879,823.250	2.318
107	8	300	60	1200	25	1,820,087.880	2.229
108	8	300	60	1200	62.5	1,715,591.130	4.117
109	8	600	20	300	10	2,415,228.750	9.607
110	8	600	20	300	25	1,793,424.630	10.807
111	8	600	20	300	62.5	1,770,259.130	12.234
112	8	600	20	600	10	3,891,424.750	17.776
113	8	600	20	600	25	2,421,775.500	11.855
114	8	600	20	600	62.5	2,327,846.750	19.771
115	8	600	20	1200	10	6,770,243.500	33.581
116	8	600	20	1200	25	3,420,722.000	10.675
117	8	600	20	1200	62.5	3,070,621.500	25.036
118	8	600	40	300	10	2,374,296.250	12.246
119	8	600	40	300	25	1,734,667.750	10.131
120	8	600	40	300	62.5	1,712,053.630	12.369
121	8	600	40	600	10	3,852,031.500	23.958
122	8	600	40	600	25	2,357,085.250	10.244
123	8	600	40	600	62.5	2,270,525.000	16.309
124	8	600	40	1200	10	6,677,481.500	41.213
125	8	600	40	1200	25	3,361,483.000	11.614
126	8	600	40	1200	62.5	2,896,619.000	22.919
127	8	600	60	300	10	2,478,868.250	10.744
128	8	600	60	300	25	1,733,622.000	11.633
129	8	600	60	300	62.5	1,700,210.500	13.537
130	8	600	60	600	10	4,091,903.000	25.565
131	8	600	60	600	25	2,364,766.500	10.342
132	8	600	60	600	62.5	2,216,887.000	18.884
133	8	600	60	1200	10	7,357,203.500	45.831
134	8	600	60	1200	25	3,490,734.750	10.707
135	8	600	60	1200	62.5	2,902,789.750	23.011
136	8	1200	20	300	10	4,148,498.500	81.936
137	8	1200	20	300	25	2,869,331.750	76.203
138	8	1200	20	300	62.5	2,819,495.250	97.658
139	8	1200	20	600	10	6,860,668.500	199.323
140	8	1200	20	600	25	3,943,211.750	72.759
141	8	1200	20	600	62.5	3,684,010.250	135.968
142	8	1200	20	1200	10	12,343,418.000	460.461
143	8	1200	20	1200	25	5,739,109.000	82.128
144	8	1200	20	1200	62.5	4,840,069.000	173.298
145	8	1200	40	300	10	4,049,488.000	113.631
146	8	1200	40	300	25	2,398,389.250	83.222
147	8	1200	40	300	62.5	2,846,925.000	101.661
148	8	1200	40	600	10	6,720,026.000	167.110
149	8	1200	40	600	25	3,922,173.250	69.803
150	8	1200	40	600	62.5	3,735,393.250	136.463

EX	# of Types	# of Locations	Seed #	R_Hubcost	R_Cap	Obj	Time
151	8	1200	40	1200	10	11,882,505.000	453.498
152	8	1200	40	1200	25	5,728,484.000	91.826
153	8	1200	40	1200	62.5	4,796,652.000	182.439
154	8	1200	60	300	10	4,122,614.000	103.129
155	8	1200	60	300	25	2,941,193.750	82.884
156	8	1200	60	300	62.5	2,899,947.750	90.330
157	8	1200	60	600	10	6,762,479.500	229.744
158	8	1200	60	600	25	4,033,366.000	78.466
159	8	1200	60	600	62.5	3,822,913.750	153.579
160	8	1200	60	1200	10	12,118,775.000	485.505
161	8	1200	60	1200	25	5,907,992.000	95.687
162	8	1200	60	1200	62.5	4,879,494.000	185.915
163	12	300	20	300	10	1,422,569.130	1.299
164	12	300	20	300	25	1,174,683.880	1.398
165	12	300	20	300	62.5	1,169,707.130	1.686
166	12	300	20	600	10	2,232,768.250	1.922
167	12	300	20	600	25	1,556,225.880	1.708
168	12	300	20	600	62.5	1,521,768.750	2.251
169	12	300	20	1200	10	3,679,094.750	3.749
170	12	300	20	1200	25	2,119,424.750	2.143
171	12	300	20	1200	62.5	1,972,698.250	3.186
172	12	300	40	300	10	1,245,022.880	1.208
173	12	300	40	300	25	1,170,244.880	1.358
174	12	300	40	300	62.5	1,170,244.880	1.533
175	12	300	40	600	10	1,903,513.880	1.531
176	12	300	40	600	25	1,573,169.380	2.102
177	12	300	40	600	62.5	1,553,434.500	2.287
178	12	300	40	1200	10	3,056,058.000	2.316
179	12	300	40	1200	25	2,111,616.250	2.227
180	12	300	40	1200	62.5	2,066,684.130	3.696
181	12	300	60	300	10	1,267,594.500	1.294
182	12	300	60	300	25	1,107,888.750	1.462
183	12	300	60	300	62.5	1,107,888.750	1.995
184	12	300	60	600	10	1,935,130.130	1.919
185	12	300	60	600	25	1,461,961.880	1.976
186	12	300	60	600	62.5	1,461,961.880	2.229
187	12	300	60	1200	10	3,011,663.750	2.479
188	12	300	60	1200	25	1,934,336.880	2.188
189	12	300	60	1200	62.5	1,902,988.250	3.216
190	12	600	20	300	10	2,210,059.500	9.929
191	12	600	20	300	25	1,863,698.630	12.480
192	12	600	20	300	62.5	1,862,705.130	12.959
193	12	600	20	600	10	3,515,978.250	13.147
194	12	600	20	600	25	2,522,754.000	14.491
195	12	600	20	600	62.5	2,420,109.500	19.129
196	12	600	20	1200	10	5,794,048.500	30.682
197	12	600	20	1200	25	3,485,482.500	15.286
198	12	600	20	1200	62.5	3,098,109.750	28.280
199	12	600	40	300	10	2,419,706.000	11.692
200	12	600	40	300	25	1,886,373.130	10.892

EX	# of Types	# of Locations	Seed #	R_Hubcost	R_Cap	Obj	Time
201	12	600	40	300	62.5	1,869,586.500	11.985
202	12	600	40	600	10	3,770,711.250	17.024
203	12	600	40	600	25	2,521,190.500	11.829
204	12	600	40	600	62.5	2,434,287.500	15.742
205	12	600	40	1200	10	6,424,507.500	32.766
206	12	600	40	1200	25	3,484,714.500	12.640
207	12	600	40	1200	62.5	3,239,986.000	25.432
208	12	600	60	300	10	2,510,865.000	11.455
209	12	600	60	300	25	1,901,206.500	13.333
210	12	600	60	300	62.5	1,867,358.500	14.326
211	12	600	60	600	10	4,047,076.500	19.166
212	12	600	60	600	25	2,577,008.000	13.522
213	12	600	60	600	62.5	2,430,793.750	19.647
214	12	600	60	1200	10	6,881,237.500	32.528
215	12	600	60	1200	25	3,649,370.000	13.178
216	12	600	60	1200	62.5	3,167,485.250	26.571
217	12	1200	20	300	10	3,937,353.750	122.024
218	12	1200	20	300	25	3,036,538.500	84.433
219	12	1200	20	300	62.5	3,023,474.500	97.986
220	12	1200	20	600	10	6,335,782.000	126.737
221	12	1200	20	600	25	4,074,255.500	86.526
222	12	1200	20	600	62.5	3,934,305.000	147.214
223	12	1200	20	1200	10	10,827,004.000	384.420
224	12	1200	20	1200	25	5,742,784.000	83.003
225	12	1200	20	1200	62.5	5,154,536.500	202.710
226	12	1200	40	300	10	3,897,252.750	62.880
227	12	1200	40	300	25	3,124,812.250	84.982
228	12	1200	40	300	62.5	3,112,074.250	94.947
229	12	1200	40	600	10	6,288,969.500	141.249
230	12	1200	40	600	25	4,156,829.750	90.287
231	12	1200	40	600	62.5	4,042,869.500	134.463
232	12	1200	40	1200	10	10,735,915.000	344.684
233	12	1200	40	1200	25	5,742,784.000	87.381
234	12	1200	40	1200	62.5	5,208,971.500	207.083
235	12	1200	60	300	10	3,971,105.500	63.806
236	12	1200	60	300	25	3,243,790.500	82.666
237	12	1200	60	300	62.5	3,221,854.000	94.919
238	12	1200	60	600	10	6,410,867.500	103.092
239	12	1200	60	600	25	4,302,685.500	88.989
240	12	1200	60	600	62.5	4,185,199.750	146.470
241	12	1200	60	1200	10	10,853,386.000	287.913
242	12	1200	60	1200	25	6,009,011.000	98.122
243	12	1200	60	1200	62.5	5,448,110.500	217.851

APPENDIX B

Tukey Tests on Four Factors versus Algorithm Time in Chapter 4

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable log(time)
All Pairwise Comparisons among Levels of Type
Type =  4  subtracted from:

Type    Lower   Center     Upper  ---------+---------+---------+-------
 8    -0.1651  -0.1136  -0.06218   (--------*--------)
12    -0.1769  -0.1255  -0.07400  (-------*--------)
                                  ---------+---------+---------+-------
                                       -0.120    -0.060     0.000


Type =  8  subtracted from:

Type     Lower    Center    Upper  ---------+---------+---------+-------
12    -0.06328  -0.01182  0.03964                   (--------*--------)
                                   ---------+---------+---------+-------
                                       -0.120    -0.060     0.000


Tukey Simultaneous Tests
Response Variable log(time)
All Pairwise Comparisons among Levels of Type
Type =  4  subtracted from:

      Difference       SE of           Adjusted
Type    of Means  Difference  T-Value   P-Value
 8       -0.1136     0.02179   -5.215    0.0000
12       -0.1255     0.02179   -5.758    0.0000


Type =  8  subtracted from:

      Difference       SE of           Adjusted
Type    of Means  Difference  T-Value   P-Value
12      -0.01182     0.02179  -0.5425    0.8504
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable log(time)
All Pairwise Comparisons among Levels of Location
Location =  300  subtracted from:

Location   Lower  Center   Upper  --+---------+---------+---------+----
 600      0.8549  0.9063  0.9578  (-*-)
1200      1.7816  1.8331  1.8846                                (-*-)
                                  --+---------+---------+---------+----
                                  0.90      1.20      1.50      1.80


Location =  600  subtracted from:

Location   Lower  Center   Upper  --+---------+---------+---------+----
1200      0.8753  0.9268  0.9782   (-*-)
                                  --+---------+---------+---------+----
                                  0.90      1.20      1.50      1.80


Tukey Simultaneous Tests
Response Variable log(time)
All Pairwise Comparisons among Levels of Location
Location =  300  subtracted from:

          Difference       SE of           Adjusted
Location    of Means  Difference  T-Value   P-Value
 600          0.9063     0.02179    41.59    0.0000
1200          1.8331     0.02179    84.13    0.0000


Location =  600  subtracted from:

          Difference       SE of           Adjusted
Location    of Means  Difference  T-Value   P-Value
1200          0.9268     0.02179    42.53    0.0000
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable log(time)
All Pairwise Comparisons among Levels of R_Capacity
R_Capacity = 10.0  subtracted from:

R_Capacity    Lower   Center    Upper     +---------+---------+---------+------
25.0        -0.3245  -0.2730  -0.2216     (--*--)
62.5        -0.1374  -0.0860  -0.0345               (---*--)
                                          +---------+---------+---------+------
                                       -0.32     -0.16      0.00      0.16


R_Capacity = 25.0  subtracted from:

R_Capacity   Lower  Center   Upper     +---------+---------+---------+------
62.5        0.1356  0.1871  0.2385                                 (---*--)
                                       +---------+---------+---------+------
                                    -0.32     -0.16      0.00      0.16


Tukey Simultaneous Tests
Response Variable log(time)
All Pairwise Comparisons among Levels of R_Capacity
R_Capacity = 10.0  subtracted from:

            Difference       SE of           Adjusted
R_Capacity    of Means  Difference  T-Value   P-Value
25.0           -0.2730     0.02179   -12.53    0.0000
62.5           -0.0860     0.02179    -3.95    0.0003


R_Capacity = 25.0  subtracted from:

            Difference       SE of           Adjusted
R_Capacity    of Means  Difference  T-Value   P-Value
62.5           0.1871     0.02179    8.584    0.0000
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable log(time)
All Pairwise Comparisons among Levels of R_HubCost
R_HubCost =  300  subtracted from:

R_HubCost    Lower  Center   Upper  --------+---------+---------+--------
 600       0.09901  0.1505  0.2019  (------*-----)
1200       0.25983  0.3113  0.3628                    (------*-----)
                                    --------+---------+---------+--------
                                          0.160     0.240     0.320


R_HubCost =  600  subtracted from:

R_HubCost   Lower  Center   Upper  --------+---------+---------+--------
1200       0.1094  0.1608  0.2123    (-----*------)
                                    --------+---------+---------+--------
                                         0.160     0.240     0.320


Tukey Simultaneous Tests
Response Variable log(time)
All Pairwise Comparisons among Levels of R_HubCost
R_HubCost =  300  subtracted from:

           Difference       SE of           Adjusted
R_HubCost    of Means  Difference  T-Value   P-Value
 600           0.1505     0.02179    6.906    0.0000
1200           0.3113     0.02179   14.286    0.0000


R_HubCost =  600  subtracted from:

           Difference       SE of           Adjusted
R_HubCost    of Means  Difference  T-Value   P-Value
1200           0.1608     0.02179    7.380    0.0000
```

APPENDIX C

Original Data on Experimental Factors versus Solution Gap in Chapter 4

EX	# of Locations	Seed #	R_Hub cost	R_Cap	Algorithm		Lingo			Solution Gap (%)	Logarithm Time	
					Obj	Time	Obj	Better Time	Best Time		Algori Time	Lingo Better Time
1	30	20	300	10	207764.40	0.016	207764.40	3	3	0.00%	3.73	6.00
2	30	20	300	15	199699.50	0.009	199078.30	3	3	0.31%	3.48	6.00
3	30	20	300	20	199699.50	0.008	199078.30	3	3	0.31%	3.43	6.00
4	30	20	600	10	318907.19	0.008	310236.60	7	11	2.79%	2.86	5.80
5	30	20	600	15	274327.31	0.016	273300.80	2	2	0.38%	3.90	6.00
6	30	20	600	20	274327.31	0.014	272202.00	3	3	0.78%	3.67	6.00
7	30	20	1200	10	458048.81	0.023	449142.50	5	5	1.98%	3.66	6.00
8	30	20	1200	15	398911.88	0.057	398549.20	5	5	0.09%	4.06	6.00
9	30	20	1200	20	382738.41	0.007	380205.00	3	3	0.67%	3.37	6.00
10	30	40	300	10	238876.63	0.170	237177.40	3	3	0.72%	4.75	6.00
11	30	40	300	15	228264.25	0.006	227911.40	3	3	0.15%	3.30	6.00
12	30	40	300	20	228264.25	0.006	227911.40	2	2	0.15%	3.48	6.00
13	30	40	600	10	364078.88	0.007	346693.20	7	7	5.01%	3.00	6.00
14	30	40	600	15	330004.41	0.007	327386.10	2	2	0.80%	3.54	6.00
15	30	40	600	20	330004.41	0.007	322119.90	2	2	2.45%	3.54	6.00
16	30	40	1200	10	530525.19	0.006	514963.10	8	17	3.02%	2.55	5.67
17	30	40	1200	15	480897.69	0.007	467389.50	5	6	2.89%	3.07	5.92
18	30	40	1200	20	455894.63	0.005	449158.80	2	2	1.50%	3.40	6.00
19	30	60	300	10	197417.13	0.008	196130.90	2	2	0.66%	3.60	6.00
20	30	60	300	15	195397.70	0.018	195397.70	3	3	0.00%	3.78	6.00
21	30	60	300	20	195397.70	0.018	195397.70	3	3	0.00%	3.78	6.00
22	30	60	600	10	294470.84	0.061	286790.00	3	3	2.68%	4.31	6.00
23	30	60	600	15	280128.88	0.014	279095.00	3	3	0.37%	3.67	6.00
24	30	60	600	20	275570.00	0.014	275570.00	3	3	0.00%	3.67	6.00
25	30	60	1200	10	432310.81	0.197	424630.00	3	4	1.81%	4.69	5.88
26	30	60	1200	15	396462.63	0.011	395394.00	3	3	0.27%	3.56	6.00
27	30	60	1200	20	388577.00	0.012	388577.00	2	2	0.00%	3.78	6.00
28	60	20	300	10	330258.63	0.058	325200.00	8	8	1.56%	3.86	6.00
29	60	20	300	15	325320.63	0.031	320169.00	7	7	1.61%	3.65	6.00
30	60	20	300	20	321592.75	0.024	317553.00	6	6	1.27%	3.60	6.00
31	60	20	600	10	464595.47	0.055	461580.00	20	25	0.65%	3.34	5.90
32	60	20	600	15	446895.00	0.030	430841.00	19	19	3.73%	3.20	6.00
33	60	20	600	20	448773.69	0.027	426348.00	5	5	5.26%	3.73	6.00
34	60	20	1200	10	756541.69	0.065	711437.00	26	628	6.34%	2.01	4.62
35	60	20	1200	15	613135.63	0.016	599092.00	16	16	2.34%	3.00	6.00
36	60	20	1200	20	583096.44	0.022	583097.00	6	6	0.00%	3.56	6.00
37	60	40	300	10	365384.06	0.197	361739.00	14	16	1.01%	4.09	5.94
38	60	40	300	15	375636.22	0.028	374489.00	10	10	0.31%	3.45	6.00
39	60	40	300	20	363073.09	0.037	353437.00	11	11	2.73%	3.53	6.00
40	60	40	600	10	576982.94	0.059	552586.00	21	27	4.42%	3.34	5.89

EX	# of Locations	Seed #	R_Hub cost	R_Cap	Algorithm		Lingo			Solution Gap (%)	Logarithm Time	
					Obj	Time	Obj	Better Time	Best Time		Algori Time	Lingo Better Time
41	60	40	600	15	536698.63	0.027	525667.00	25	40	2.10%	2.83	5.80
42	60	40	600	20	490258.28	0.022	475974.00	13	13	3.00%	3.23	6.00
43	60	40	1200	10	909921.50	0.037	859141.00	24	94	5.91%	2.60	5.41
44	60	40	1200	15	823938.75	0.024	760436.00	31	978	8.35%	1.39	4.50
45	60	40	1200	20	706180.56	0.024	683344.00	19	57	3.34%	2.62	5.52
46	60	60	300	10	342885.03	0.030	340003.00	8	8	0.85%	3.57	6.00
47	60	60	300	15	332860.00	0.023	332860.00	6	6	0.00%	3.58	6.00
48	60	60	300	20	332860.00	0.025	332860.00	6	6	0.00%	3.62	6.00
49	60	60	600	10	513624.00	0.026	509005.00	15	16	0.91%	3.21	5.97
50	60	60	600	15	479004.03	0.023	472620.00	13	15	1.35%	3.19	5.94
51	60	60	600	20	457316.47	0.021	456422.00	6	6	0.20%	3.54	6.00
52	60	60	1200	10	815156.06	0.024	775420.00	24	63	5.12%	2.58	5.58
53	60	60	1200	15	660283.13	0.025	648452.00	13	21	1.82%	3.08	5.79
54	60	60	1200	20	640477.00	0.029	625789.00	17	23	2.35%	3.10	5.87
55	90	20	300	10	519639.53	0.362	502987.00	65	156	3.31%	3.37	5.62
56	90	20	300	15	471392.13	0.057	460477.00	24	26	2.37%	3.34	5.97
57	90	20	300	20	462598.00	0.055	453272.00	13	13	2.06%	3.63	6.00
58	90	20	600	10	731492.50	0.229	712114.00	95	1286	2.72%	2.25	4.87
59	90	20	600	15	617615.81	0.043	612819.00	40	48	0.78%	2.95	5.92
60	90	20	600	20	597467.13	0.044	590773.00	14	14	1.13%	3.50	6.00
61	90	20	1200	10	1111459.38	0.294	1071939.00	110	106885	3.69%	0.44	3.01
62	90	20	1200	15	874570.38	0.058	854487.10	68	109	2.35%	2.73	5.80
63	90	20	1200	20	825661.63	0.051	804843.80	28	34	2.59%	3.18	5.92
64	90	40	300	10	531687.13	0.163	512050.00	48	1024	3.84%	2.20	4.67
65	90	40	300	15	451588.53	0.275	445184.10	50	54	1.44%	3.71	5.97
66	90	40	300	20	451345.53	0.066	441181.10	14	14	2.30%	3.67	6.00
67	90	40	600	10	815581.75	0.452	797926.00	70	14088	2.21%	1.51	3.70
68	90	40	600	15	673235.19	0.165	642335.80	89	306	4.81%	2.73	5.46
69	90	40	600	20	617103.38	0.048	608470.40	33	39	1.42%	3.09	5.93
70	90	40	1200	10	1324506.75	0.181	1211310.00	97	97098	9.34%	0.27	3.00
71	90	40	1200	15	1020712.81	0.048	954932.50	108	9462	6.89%	0.71	4.06
72	90	40	1200	20	856337.50	0.043	852685.00	36	66	0.43%	2.81	5.74
73	90	60	300	10	553407.63	0.215	521291.00	66	357	6.16%	2.78	5.27
74	90	60	300	15	449228.59	0.100	448152.30	26	29	0.24%	3.54	5.95
75	90	60	300	20	448499.31	0.101	443178.30	13	13	1.20%	3.89	6.00
76	90	60	600	10	834707.06	0.133	784363.70	82	555	6.42%	2.38	5.17
77	90	60	600	15	645247.00	0.089	640636.90	24	72	0.72%	3.09	5.52
78	90	60	600	20	613902.44	0.040	611963.50	13	13	0.32%	3.49	6.00
79	90	60	1200	10	1341766.25	0.286	1231652.00	119	38496	8.94%	0.87	3.49
80	90	60	1200	15	983864.06	0.039	962923.10	31	120	2.17%	2.51	5.41
81	90	60	1200	20	899719.44	0.046	882357.90	41	76	1.97%	2.78	5.73

APPENDIX D

Tukey Test on Three Factors versus Solution Gap in Chapter 4

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable solution gap
All Pairwise Comparisons among Levels of Location
Location = 30  subtracted from:

Location     Lower   Center    Upper  ------+---------+---------+---------+
60        0.002312  0.01361  0.02490          (-----------*----------)
90        0.007975  0.01927  0.03057                (----------*-----------)
                                     ------+---------+---------+---------+
                                         0.000     0.010     0.020     0.030


Location = 60  subtracted from:

Location      Lower    Center    Upper  ------+---------+---------+---------+
90        -0.005632  0.005663  0.01696  (-----------*----------)
                                      ------+---------+---------+---------+
                                          0.000     0.010     0.020     0.030


Tukey Simultaneous Tests
Response Variable solution gap
All Pairwise Comparisons among Levels of Location
Location = 30  subtracted from:

          Difference       SE of           Adjusted
Location    of Means  Difference  T-Value   P-Value
60           0.01361    0.004726    2.879    0.0143
90           0.01927    0.004726    4.078    0.0003


Location = 60  subtracted from:

          Difference       SE of           Adjusted
Location    of Means  Difference  T-Value   P-Value
90           0.005663   0.004726    1.198    0.4580
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable solution gap
All Pairwise Comparisons among Levels of R_Capacity
R_Capacity = 10  subtracted from:

R_Capacity     Lower    Center      Upper
15          -0.02738  -0.01609  -0.004790
20          -0.03153  -0.02024  -0.008942

R_Capacity  ------+---------+---------+---------+
15             (---------*--------)
20          (--------*---------)
            ------+---------+---------+---------+
               -0.024    -0.012    -0.000     0.012


R_Capacity = 15  subtracted from:

R_Capacity     Lower     Center     Upper
20          -0.01545  -0.004152  0.007143

R_Capacity  ------+---------+---------+---------+
20                    (---------*--------)
            ------+---------+---------+---------+
               -0.024    -0.012    -0.000     0.012


Tukey Simultaneous Tests
Response Variable solution gap
All Pairwise Comparisons among Levels of R_Capacity
R_Capacity = 10  subtracted from:

            Difference       SE of           Adjusted
R_Capacity    of Means  Difference  T-Value   P-Value
15             -0.01609    0.004726   -3.404    0.0031
20             -0.02024    0.004726   -4.282    0.0002


R_Capacity = 15  subtracted from:

            Difference       SE of           Adjusted
R_Capacity    of Means  Difference  T-Value   P-Value
20            -0.004152    0.004726  -0.8785    0.6555
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable solution gap
All Pairwise Comparisons among Levels of R_HubCost
R_HubCost =  300  subtracted from:

R_HubCost      Lower    Center    Upper  ---+---------+---------+---------+---
 600        -0.002832  0.008463  0.01976  (----------*----------)
1200         0.007820  0.019115  0.03041            (----------*----------)
                                          ---+---------+---------+---------+---
                                           0.000     0.010     0.020     0.030


R_HubCost =  600  subtracted from:

R_HubCost      Lower   Center    Upper  ---+---------+---------+---------+---
1200        -0.000643  0.01065  0.02195    (----------*----------)
                                         ---+---------+---------+---------+---
                                          0.000     0.010     0.020     0.030


Tukey Simultaneous Tests
Response Variable solution gap
All Pairwise Comparisons among Levels of R_HubCost
R_HubCost =  300  subtracted from:

           Difference       SE of           Adjusted
R_HubCost    of Means  Difference  T-Value   P-Value
 600         0.008463    0.004726    1.791    0.1798
1200         0.019115    0.004726    4.045    0.0004


R_HubCost =  600  subtracted from:

           Difference       SE of           Adjusted
R_HubCost    of Means  Difference  T-Value   P-Value
1200          0.01065    0.004726    2.254    0.0689
```

APPENDIX E

Original Data of Inter-pool Moves and Asset Replacement

on Experimental Factors versus Solution Gap in Chapter 5

EX	# of Types	# of Ages / # of Times	# of Pools	Seed #	Algorithm		Branch and Bound		Solution Gap(%)	# of Integer Var	# of Constraints
					Obj	Time	Obj	Time			
1	4	4	10	20	166177122	0.102	161768300	2	2.73%	7200	801
2	4	4	10	40	174664867	0.094	172059700	3	1.51%	7200	801
3	4	4	10	60	151305511	0.109	146165300	2	3.52%	7200	801
4	4	4	20	20	321566665	0.203	311883400	18	3.10%	27200	1601
5	4	4	20	40	317531012	0.203	309427800	16	2.62%	27200	1601
6	4	4	20	60	359055254	0.188	351431600	16	2.17%	27200	1601
7	4	4	30	20	486966866	0.312	472834500	51	2.99%	60000	2401
8	4	4	30	40	452234865	0.296	438719200	50	3.08%	60000	2401
9	4	4	30	60	472342392	0.312	456040400	51	3.57%	60000	2401
10	4	5	10	20	241868544	0.123	238688700	4	1.33%	11200	1201
11	4	5	10	40	237714073	0.125	235078400	4	1.12%	11200	1201
12	4	5	10	60	222220511	0.109	219165100	4	1.39%	11200	1201
13	4	5	20	20	443701870	0.249	437366400	31	1.45%	42400	2401
14	4	5	20	40	488453418	0.250	481641000	31	1.41%	42400	2401
15	4	5	20	60	412053386	0.249	403249400	31	2.18%	42400	2401
16	4	5	30	20	655410771	0.390	646589800	102	1.36%	93600	3601
17	4	5	30	40	686451394	0.375	678531100	103	1.17%	93600	3601
18	4	5	30	60	676298723	0.343	669210600	102	1.06%	93600	3601
19	4	6	10	20	305800201	0.125	304209700	7	0.52%	16080	1681
20	4	6	10	40	253983859	0.156	250996500	8	1.19%	16080	1681
21	4	6	10	60	253037981	0.185	250595700	7	0.97%	16080	1681
22	4	6	20	20	592708572	0.281	588150900	54	0.77%	60960	3361
23	4	6	20	40	581302890	0.265	577255700	55	0.70%	60960	3361
24	4	6	20	60	531311832	0.296	527780200	55	0.67%	60960	3361
25	4	6	30	20	822403811	0.421	814782200	182	0.94%	134640	5041
26	4	6	30	40	807894151	0.452	800560700	181	0.92%	134640	5041
27	4	6	30	60	890299895	0.452	883071200	180	0.82%	134640	5041
28	5	4	10	20	329025480	0.109	321903300	4	2.21%	9000	1001
29	5	4	10	40	370318477	0.131	361880400	4	2.33%	9000	1001
30	5	4	10	60	278725225	0.141	256276000	3	8.76%	9000	1001
31	5	4	20	20	541241862	0.266	514730000	25	5.15%	34000	2001
32	5	4	20	40	632329782	0.234	619217600	25	2.12%	34000	2001
33	5	4	20	60	608190350	0.250	588864900	25	3.28%	34000	2001
34	5	4	30	20	894654562	0.390	859611300	80	4.08%	75000	3001
35	5	4	30	40	931933260	0.421	897943000	80	3.79%	75000	3001
36	5	4	30	60	1004438577	0.390	976621000	80	2.85%	75000	3001
37	5	5	10	20	451089305	0.141	444283900	6	1.53%	14000	1501
38	5	5	10	40	416242556	0.182	409694400	7	1.60%	14000	1501
39	5	5	10	60	417904645	0.141	412482200	7	1.31%	14000	1501
40	5	5	20	20	836824263	0.296	824138500	49	1.54%	53000	3001

EX	# of Types	# of Ages / # of Times	# of Pools	Seed #	Algorithm		Branch and Bound		Solution Gap(%)	# of Integer Var	# of Constraints
					Obj	Time	Obj	Time			
41	5	5	20	40	812180370	0.312	800301400	48	1.48%	53000	3001
42	5	5	20	60	873690801	0.297	861711900	49	1.39%	53000	3001
43	5	5	30	20	1173578267	0.484	1157793000	162	1.36%	117000	4501
44	5	5	30	40	1299958359	0.483	1279813000	163	1.57%	117000	4501
45	5	5	30	60	1291660430	0.468	1278317000	163	1.04%	117000	4501
46	5	6	10	20	525106422	0.172	517481700	12	1.47%	20100	2101
47	5	6	10	40	469905923	0.187	467213900	11	0.58%	20100	2101
48	5	6	10	60	584828000	0.156	580382300	12	0.77%	20100	2101
49	5	6	20	20	966982224	0.359	953773700	84	1.38%	76200	4201
50	5	6	20	40	1030603578	0.359	1019497000	84	1.09%	76200	4201
51	5	6	20	60	1007403852	0.343	1000459000	84	0.69%	76200	4201
52	5	6	30	20	1520003143	0.577	1505092000	278	0.99%	168300	6301
53	5	6	30	40	1634206173	0.546	1625960000	280	0.51%	168300	6301
54	5	6	30	60	1423337738	0.577	1409902000	280	0.95%	168300	6301
55	6	4	10	20	624933770	0.138	608119400	5	2.76%	10800	1201
56	6	4	10	40	635881361	0.125	617064400	5	3.05%	10800	1201
57	6	4	10	60	587926156	0.150	574143600	6	2.40%	10800	1201
58	6	4	20	20	1061738503	0.280	1015290000	36	4.57%	40800	2401
59	6	4	20	40	1201476903	0.312	1147053000	36	4.74%	40800	2401
60	6	4	20	60	1154576744	0.312	1129473000	37	2.22%	40800	2401
61	6	4	30	20	1755402235	0.452	1698052000	119	3.38%	90000	3601
62	6	4	30	40	1724215748	0.436	1657628000	114	4.02%	90000	3601
63	6	4	30	60	1730211315	0.500	1676334000	115	3.21%	90000	3601
64	6	5	10	20	749002718	0.193	733473500	10	2.12%	16800	1801
65	6	5	10	40	863929959	0.171	856548000	10	0.86%	16800	1801
66	6	5	10	60	769973159	0.202	757624600	9	1.63%	16800	1801
67	6	5	20	20	1476566889	0.358	1446483000	72	2.08%	63600	3601
68	6	5	20	40	1693027994	0.359	1672478000	73	1.23%	63600	3601
69	6	5	20	60	1594997420	0.359	1577586000	71	1.10%	63600	3601
70	6	5	30	20	2175945263	0.593	2114587000	236	2.90%	140400	5401
71	6	5	30	40	2093817719	0.561	2070215000	237	1.14%	140400	5401
72	6	5	30	60	2348164646	0.577	2316510000	239	1.37%	140400	5401
73	6	6	10	20	974665527	0.203	967520800	17	0.74%	24120	2521
74	6	6	10	40	1106199489	0.217	1100371000	17	0.53%	24120	2521
75	6	6	10	60	936282618	0.218	930710700	16	0.60%	24120	2521
76	6	6	20	20	2144049441	0.437	2126809000	127	0.81%	91440	5041
77	6	6	20	40	1760218799	0.437	1740476000	126	1.13%	91440	5041
78	6	6	20	60	2226210019	0.421	2215181000	122	0.50%	91440	5041
79	6	6	30	20	3222033315	0.671	3204782000	437	0.54%	201960	7561
80	6	6	30	40	2797069691	0.748	2774747000	430	0.80%	201960	7561
81	6	6	30	60	3055006311	0.718	3037558000	420	0.57%	201960	7561

APPENDIX F

Tukey Tests of Inter-pool Moves and Asset Replacement

on the Number of Car Ages/ Seasonal Periods versus Solution Gap in Chapter 5

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable solution gap
All Pairwise Comparisons among Levels of age/season
age/season = 4  subtracted from:

age/season     Lower    Center     Upper
5           -0.02426  -0.01869  -0.01312
6           -0.03078  -0.02521  -0.01964

age/season    -+---------+---------+---------+-----
5                   (----*-----)
6             (-----*----)
              -+---------+---------+---------+-----
            -0.030    -0.020    -0.010     0.000


age/season = 5  subtracted from:

age/season     Lower     Center      Upper
6           -0.01209  -0.006514  -0.000943

age/season    -+---------+---------+---------+-----
6                            (----*-----)
              -+---------+---------+---------+-----
            -0.030    -0.020    -0.010     0.000


Tukey Simultaneous Tests
Response Variable solution gap
All Pairwise Comparisons among Levels of age/season
age/season = 4  subtracted from:

            Difference       SE of           Adjusted
age/season    of Means  Difference  T-Value   P-Value
5             -0.01869    0.002331    -8.02    0.0000
6             -0.02521    0.002331   -10.81    0.0000


age/season = 5  subtracted from:

            Difference       SE of           Adjusted
age/season    of Means  Difference  T-Value   P-Value
6            -0.006514    0.002331   -2.794    0.0180
```

APPENDIX G

Tukey Tests of Inter-pool Moves and Asset Replacement

on Three Factors versus Computing Time in Chapter 5

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable algorithm time
All Pairwise Comparisons among Levels of Type
Type = 4  subtracted from:

Type    Lower   Center    Upper  ---------+---------+---------+-------
5     0.03816  0.06470  0.09125  (------*-------)
6     0.10245  0.12900  0.15555                   (-------*------)
                                ---------+---------+---------+-------
                                       0.070     0.105     0.140


Type = 5  subtracted from:

Type    Lower   Center    Upper  ---------+---------+---------+-------
6     0.03775  0.06430  0.09084  (------*-------)
                                ---------+---------+---------+-------
                                       0.070     0.105     0.140


Tukey Simultaneous Tests
Response Variable algorithm time
All Pairwise Comparisons among Levels of Type
Type = 4  subtracted from:

      Difference       SE of           Adjusted
Type    of Means  Difference  T-Value   P-Value
5        0.06470     0.01111    5.826    0.0000
6        0.12900     0.01111   11.614    0.0000


Type = 5  subtracted from:

      Difference       SE of           Adjusted
Type    of Means  Difference  T-Value   P-Value
6        0.06430     0.01111    5.789    0.0000
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable algorithm time
All Pairwise Comparisons among Levels of age/season
age/season = 4  subtracted from:

age/season    Lower   Center    Upper   -+---------+---------+---------+-----
5           0.03027  0.05681  0.08336   (------*-------)
6           0.08912  0.11567  0.14221                  (-------*-------)
                                        -+---------+---------+---------+-----
                                       0.035     0.070     0.105     0.140


age/season = 5  subtracted from:

age/season    Lower   Center    Upper   -+---------+---------+---------+-----
6           0.03231  0.05885  0.08540   (-------*------)
                                        -+---------+---------+---------+-----
                                       0.035     0.070     0.105     0.140


Tukey Simultaneous Tests
Response Variable algorithm time
All Pairwise Comparisons among Levels of age/season
age/season = 4  subtracted from:

            Difference       SE of           Adjusted
age/season    of Means  Difference  T-Value   P-Value
5              0.05681     0.01111    5.115    0.0000
6              0.11567     0.01111   10.414    0.0000


age/season = 5  subtracted from:

            Difference       SE of           Adjusted
age/season    of Means  Difference  T-Value   P-Value
6              0.05885     0.01111    5.299    0.0000
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable algorithm time
All Pairwise Comparisons among Levels of pool
pool = 10  subtracted from:

pool   Lower  Center   Upper  --+---------+---------+---------+----
20    0.1242  0.1507  0.1773  (---*--)
30    0.3009  0.3274  0.3540                          (---*---)
                              --+---------+---------+---------+----
                              0.140     0.210     0.280     0.350


pool = 20  subtracted from:

pool   Lower  Center   Upper  --+---------+---------+---------+----
30    0.1501  0.1767  0.2032     (---*---)
                              --+---------+---------+---------+----
                              0.140     0.210     0.280     0.350


Tukey Simultaneous Tests
Response Variable algorithm time
All Pairwise Comparisons among Levels of pool
pool = 10  subtracted from:

      Difference       SE of           Adjusted
pool    of Means  Difference  T-Value   P-Value
20        0.1507     0.01111    13.57    0.0000
30        0.3274     0.01111    29.48    0.0000


pool = 20  subtracted from:

      Difference       SE of           Adjusted
pool    of Means  Difference  T-Value   P-Value
30        0.1767     0.01111    15.91    0.0000
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable lingo time
All Pairwise Comparisons among Levels of Type
Type = 4  subtracted from:

Type   Lower  Center  Upper  -+---------+---------+---------+-----
5     -4.363   27.96  60.29  (---------*----------)
6     34.044   66.37  98.70             (----------*----------)
                            -+---------+---------+---------+-----
                             0        30        60        90


Type = 5  subtracted from:

Type  Lower  Center  Upper  -+---------+---------+---------+-----
6     6.081   38.41  70.73     (----------*----------)
                            -+---------+---------+---------+-----
                             0        30        60        90


Tukey Simultaneous Tests
Response Variable lingo time
All Pairwise Comparisons among Levels of Type
Type = 4  subtracted from:

      Difference       SE of           Adjusted
Type    of Means  Difference  T-Value   P-Value
5          27.96       13.53    2.067    0.1037
6          66.37       13.53    4.907    0.0000


Type = 5  subtracted from:

      Difference       SE of           Adjusted
Type    of Means  Difference  T-Value   P-Value
6          38.41       13.53    2.840    0.0159
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable lingo time
All Pairwise Comparisons among Levels of age/season
age/season = 4  subtracted from:

age/season   Lower  Center   Upper  --------+---------+---------+--------
5            5.266   37.59   69.92  (--------*--------)
6           62.415   94.74  127.07                  (--------*--------)
                                    --------+---------+---------+--------
                                           35        70       105


age/season = 5  subtracted from:

age/season  Lower  Center  Upper  --------+---------+---------+--------
6           24.82   57.15  89.47       (--------*---------)
                                   --------+---------+---------+--------
                                          35        70       105


Tukey Simultaneous Tests
Response Variable lingo time
All Pairwise Comparisons among Levels of age/season
age/season = 4  subtracted from:

            Difference       SE of           Adjusted
age/season    of Means  Difference  T-Value   P-Value
5                37.59       13.53    2.779    0.0187
6                94.74       13.53    7.005    0.0000


age/season = 5  subtracted from:

            Difference       SE of           Adjusted
age/season    of Means  Difference  T-Value   P-Value
6                57.15       13.53    4.225    0.0002
```

```
Tukey 95.0% Simultaneous Confidence Intervals
Response Variable lingo time
All Pairwise Comparisons among Levels of pool
pool = 10  subtracted from:

pool   Lower  Center   Upper  -------+---------+---------+---------
20     15.01   47.33   79.66  (----*----)
30    142.23  174.56  206.88                      (----*----)
                             -------+---------+---------+---------
                                   60       120       180


pool = 20  subtracted from:

pool  Lower  Center  Upper  -------+---------+---------+---------
30    94.90   127.2  159.5              (----*-----)
                           -------+---------+---------+---------
                                 60       120       180


Tukey Simultaneous Tests
Response Variable lingo time
All Pairwise Comparisons among Levels of pool
pool = 10  subtracted from:

      Difference       SE of           Adjusted
pool    of Means  Difference  T-Value   P-Value
20         47.33       13.53    3.500    0.0023
30        174.56       13.53   12.906    0.0000


pool = 20  subtracted from:

      Difference       SE of           Adjusted
pool    of Means  Difference  T-Value   P-Value
30         127.2       13.53    9.406    0.0000
```

Vita

Gen-Han Wu

Gen-Han Wu was born on December 21, 1973, in Tainan, Taiwan, and lived there until finishing high school. In June 1997, he received a B.S. degree in Industrial Engineering from the National Tsing Hua University, Taiwan. He also obtained a M.S. degree in Industrial Engineering and Engineering Management from the National Tsing Hua University, Taiwan in July 2000. Then, he worked for Wistron Corporation as a senior industrial engineer, conducting a series of desktop computer assemblies. In April 2002, he joined United Microelectronic Corp. (UMC) as a production controller, where he conducted the capacity planning and new product implementation of a 300mm semiconductor wafer foundry. In January 2004, Mr. Wu joined the Department of Industrial and Manufacturing Engineering at the Pennsylvania State University as a Ph.D. student. His research interests include facility location, applied operations research, meta-heuristics, and service enterprise optimization. He has presented his research at the PSU engineering research symposium, the MOPTA, the CORS-INFORMS, the INFORMS Annual Meeting and served as a session chair at INFORMS Annual Meeting. His full paper submitted to the APIEMS Conference in December, 2009 has been accepted. He is also a member of INFORMS.