The Dissertation Committee for Yong Jin Kim Certifies that this is the approved version of the following dissertation:

HYBRID APPROACHES TO SOLVE DYNAMIC FLEET MANAGEMENT PROBLEMS

Committee:

Hani S. Mahmassani, Supervisor

Patrick Jaillet, Co-Supervisor

C. Michael Walton

Gang Yu

Lance Manuel

HYBRID APPROACHES TO SOLVE DYNAMIC FLEET MANAGEMENT PROBLEMS

by

Yong Jin Kim, BS, MS

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

August 2003

Acknowledgements

I would like to express my utmost gratitude to my dissertation supervisors, Hani S. Mahmassani and Patrick Jaillet. I owe an inestimable debt to them for their patience, invaluable comments and support. They showed me a It is my honor to have the opportunity to work with them. I would like to extend special acknowledgment to my committee members, Professors C. Michael Walton, Gang Yu and Lance Manuel for their participation and assistance. Appreciation also goes to Rebecca A. Weaver-Gill and Linda Pannell, Kathleen Rose for their collaboration with the administrative workload. I would also like to recognize graduate advisor, Professor Howard Liljestrand for his kind support.

Several people deserve special recognition. Fait Jordan Ludders must be recognized for her help to write the dissertation and friendship. My friends, Jian Yang, Iris Lin, Miguel Figliozzi, Ricardo Giesen, Ahmed Abdelghany, Khaled Abdelghany, Yi-Chang Chiu, Michael Hunter, Karthik Srinivasan and Nathan (Nhan) Huynh will always be dearly remembered. I would like to express thanks to Chungwon Lee, Jihyeon Song, Ji-Hyang Kweon, Seungwon Won, Minyoung Park, Sanghoon Lee, Chul Seo, Young-Joon Kweon, Hyun-Joon Shin, Yeon-Joo Min, Yoonsuck Choe and Jongbin Lee for their caring friendship and encouragement throughout my time at the University of Texas at Austin.

In addition, I would like to acknowledge the unconditional support and firm belief of my parents. Finally, I would like to express my eternal love to my wife, Jiyeon Kim and our son, D. Hojoong Kim.

Hybrid Approaches to Solve Dynamic Fleet Management Problems

Yong Jin Kim, PhD.

The University of Texas at Austin, 2003

Supervisors: Hani S. Mahmassani, Patrick Jaillet

The growing demand for customer-responsive, made-to-order manufacturing is stimulating the need for improved dynamic decision-making processes in commercial fleet operations. Moreover, the rapid growth of electronic commerce through the Internet is also requiring advanced and precise real-time operation of vehicle fleets. Accompanying these demand side developments/pressures, the growing availability of technologies such as AVL (Automatic Vehicle Location) systems and continuous two-way communication devices is driving developments on the supply side. These technologies enable the dispatcher to identify the current location of trucks and to communicate with drivers in real time affording the carrier fleet dispatcher the opportunity to dynamically respond to changes in demand, driver and vehicle availability, as well as traffic network conditions.

This research investigates key aspects of real-time dynamic routing and scheduling problems in fleet operation particularly in a truckload pickup-and-

delivery problem under various settings, in which information of stochastic demands is revealed on a continuous basis, i.e., as the scheduled routes are executed. The most promising solution strategies for dealing with this real-time problem are analyzed and integrated. Furthermore, this research develops, analyzes, and implements hybrid algorithms for solving them, which combine fast local heuristic approach with an optimization-based approach. Simulation experiments are developed and conducted to evaluate the performance of these algorithms.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

1.1 MOTIVATION

The trucking industry comprises a large portion of the economy. In the USA, revenue from the trucking industry is nearly five percent of the Gross Domestic Product (GDP). Furthermore, approximately 81 percent of shipping costs were attributed to the trucking industry in 1997 (http://www.fhwa.dot.gov/freightplanning/weart.htm). An estimate based on the Commodity Flow Survey shows that trucks (single mode excluding multiple modes cases) carried approximately $4.98 trillion worth of goods over 1,023 billion ton-miles in 1997 (http://www.bts.gov/ntda/cfs/97tcf-us.pdf). Based on these statistics, it is understandable that even a slight improvement in the operating efficiency of the trucking industry would greatly contribute to the overall economy. Furthermore, the current technology and economic environment in both supply and demand motivate the improvement of efficiency in the trucking industry.

On the demand side, the motivation to improve stems from a growing demand for the kind of customer-responsive, made-to-order manufacturing that has been a major factor in the global success of the likes of Dell Computer Corporation. This trend is shifting the logistics and transportation process from relying on long-planned lead times to an extremely dynamic short-term process. An indicator illustrating this trend is the inventory to retail sales ratio over the total business. The ratio was 1.53 in January 1993, 1.44 in January 1998 and had

decreased to 1.37 in December 2002 (http://www.census.gov/mtis/www/current.html).

Furthermore, according to the US Census Bureau data, E-commerce retail sales, which is defined as "the sales of goods and services where an order is placed by the buyer or price and terms of sale are negotiated over an Internet, extranet, Electronic Data Interchange (EDI) network, electronic mail, or other online system. Payment may or may not be made online (http://www.census.gov/mrts/www/current.html)", increase from 5,481 million dollars (0.7% out of total retail sales) in 1999, 4th quarter to 14,334 million dollars (1.6%) in 2002, 4th quarter. Explosive growth in the electronic commerce and the development of virtually integrated supply chains through the Internet require advanced and precise real-time operation of vehicle fleets in freight transportation systems.

Accompanying these developments on the demand side are equally compelling developments on the supply side, driven in part by the growing availability of advanced technologies such as Automatic Vehicle Location (AVL) systems and continuous two-way communication devices. These systems enable the dispatcher to identify the current location and status of trucks and to communicate with drivers in real-time. With these technologies, the carrier fleet dispatcher can dynamically respond to stochastically requested demand, changes in demand, driver and vehicle availability, as well as traffic network conditions. While real-time information allows system operators to make on-line decisions on a continuing basis to optimize performance, the manner in which this information is used will critically determine its effectiveness; inappropriate use of

information, in conjunction with incorrectly formulated models and inadequate algorithms, could result in worse performance.

While adoption of commercial real-time location-communication systems by motor carriers has been growing rapidly, the full potential of these technologies to optimize load assignment, vehicle routing and scheduling, and overall fleet logistics operations remains vastly underutilized. Surveys performed under a University of Texas at Austin project and by University of California-Irvine researchers (Regan & Golob 1999) reveal that almost 60% of major-carrier trucking fleets are equipped with some form of real-time location-communication device. However, only a very small fraction is actually using this information as part of a formal optimization procedure to support fleet operation decisions. To date, methodological developments that explicitly address on-line fleet decisions under real-time information remain in their early stages of development.

1.2 PROBLEM STATEMENT

The principal focus of this thesis is to find good and computationally efficient ways in which a commercial vehicle fleet operations decision maker, or dispatcher, can take advantage of real-time information such as current vehicle location and status, and two-way communication technologies, to dynamically manage available resources to serve the general pattern of stochastic, time-sensitive customer truckload pickup-and-delivery demands.

In this study, it is assumed that demand information including origin, destination, and service time-windows (and/or demand type) of the demand, is

revealed dynamically, i.e. the demand requests arrive at the carrier on a continuous basis. In trucking operations, the fraction of demands that carriers know in advance is typically limited. The fraction may vary widely by company, regional orientation, and the sectors of the economy in which their primary shippers tend to operate. For instance, Powell (1988) reports that sixty percent of a given day's loads at a large national carrier may be accepted on the same day that they are moved. Throughout the thesis, it is assumed that 100 percent of the demand information is revealed dynamically. The customer has his/her own desirable time frame called time-window, within which the demand should be served. It is assumed that time-window width of demands is relatively wide enough, as compared to the haul-length, to be able to construct routing schedule with multiple legs.

A dispatcher, as a decision maker, has the responsibility to make a decision on the activities of trucks in the fleet. First, the dispatcher should be able to determine the feasibility and desirability of a particular load request (acceptance/rejection decision) and inform the customer (shipper) of the decision in near real-time, or within a short time after receiving the request. This decision may have an impact on the ability to accept future demands. Second, the dispatcher must make a feasible routing schedule of the fleet of trucks, which includes assignment of the newly requested demand to a driver and possible reassignment of the existing accepted-but-not-served demands to other vehicles. Assignment of a requested load to a vehicle is not a permanent decision due to the reassignment possibility and is not necessarily communicated to a driver until

he/she is ready to travel to pick up the assigned demand. The above two decisions are tightly coupled in that each decision has significant impact on the other decision.

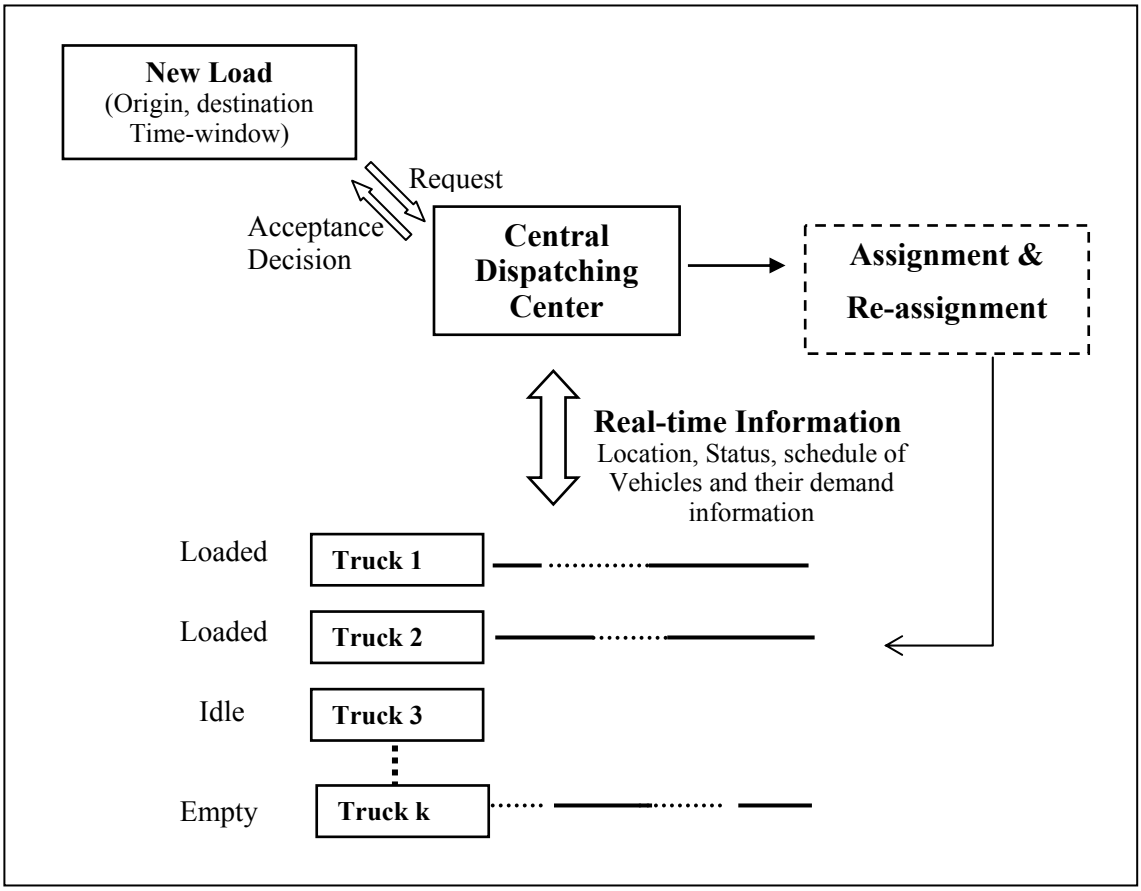


Figure 1.1 Conceptual diagram of real time decision process

Figure1.1 depicts the conceptual diagram of the real-time decision process of a dispatcher, where solid line represents the loaded movement and the dotted line represents the empty movement of the trucks.

Based on this problem setting, the objective of the decision maker is to maximize the profit by proper assignment and acceptance/rejection decisions on the dynamically realized demands. This problem is classified as a dynamic VRP (Vehicle Routing Problem) in the sense that the decision-maker does not have complete information on the demands in advance, and the input information is updated either during the execution of the route, or during the execution of the solution algorithm. In this problem, a decision at a given instant has a significant impact on the status of vehicles at subsequent decision instant, as well as on the overall performance of the system to serve dynamically requested demands.

1.3 RESEARCH CONTEXT AND GENERAL APPROACH

Routing and scheduling problems are known to be notorious *NP-Hard* problems, for which there is no guarantee of a polynomial time solution algorithm. Hence, the problem complexity is an impediment for a decision maker seeking an optimal solution in a short time even for a static version of the problem. Furthermore, in a dynamic environment, the decision-maker has limited information for demands at a decision moment and the information is updated with newly requested demands, service completion, changes in driver and vehicle availability and modified traffic network conditions.

There are two main categories of solution approaches to solve these dynamic problems. First, stochastic methods explicitly incorporate uncertainty of future demands assuming some probabilistic models of the future. These methods, however, have a limitation of problem size due to the nature of the stochastic

models; state space grows quickly with problem size (Ichoua et al. 2000). Therefore, most of the existing stochastic methods employ simple pure assignment models (or vehicle allocation models) as a base model and use aggregated time and space, for examples, Frantzeskakis and Powell (1990) and Powell (1986b, 1987, 1988). Second, the algorithms for static problems are adopted in order to solve these dynamic problems. In this case, static algorithms are repeatedly applied according to the dynamically revealed demand information. This approach is relatively easy to implement and can deal with relatively large size problems.

The choice between above two approaches should take into account the demand nature, particularly time-window width. For a time sensitive long-haul (as compared to the time-window duration of the requested demands) delivery service, the stochastic programming approaches may be appropriate. This is because long service duration reduces the flexibility of assignment. In this case, once a demand is assigned to a vehicle, it is hard to reassign it to another vehicle. Furthermore, if the time-window is tight (the specified latest pickup time is close to the demand arrival time), immediate assignment is required. In contrast, the short-haul with wide time-window demands allow a dispatcher to construct routing schedules with multiple legs. In other words, the dispatcher can hold the demands in the service queue. Moreover, adaptive modification of these routes according to the updated information may lead to great benefit. Therefore, in this thesis, whenever updated information occurs, static algorithms are applied repeatedly to solve a local snapshot problem (deterministic problem assuming no

future demand) as close to optimality as possible. Note that, the solution of a local problem is a routing schedule of the fleet of vehicles (and acceptance decision on a newly requested demand) and the optimality addressed here denotes the myopic (local) optimality rather than the global or hindsight (after the fact) optimality.

Even these local snapshot problems encountered at every decision epoch may require extremely long computation time in order to obtain the optimal solution. However, the dynamic nature of the problem necessitates short computation time. First, the dispatcher needs to provide an acceptance/rejection decision to the shipper in a short time. Second, the long computation time of the solution procedure may cause a conflict between the schedule obtained from the solution and the updated vehicle status. Finally, new demand requests may arrive during the solution procedure so that a dispatcher may lose an opportunity to utilize the updated information.

If the decision maker does not pursuit the optimal solution for the local problems, there are alternative heuristic algorithms, which can provide 'good' feasible solution for the snapshot routing and scheduling problems in a relatively short time while not guaranteeing the optimal solution. The basic approach to solve the dynamic problems, in this thesis, is to develop hybrid solution approaches, using efficient heuristic rules for a quick response to the customer while generating an initial feasible routing schedule combined with optimization-based procedures seeking the optimal solution, which are designed to improve system efficiency within the time requirements of the dynamic problem.

In this procedure, it is required to develop dynamic operational policies determining how to identify the local snapshot problems, how to apply the algorithms, and how to apply the solutions obtained from the problems to the dynamic operation of the fleet. This is because, execution time of any algorithm guaranteeing a feasible solution is not short enough to avoid all the conflicts stated above. Furthermore, the timing of the next demand arrival is not known to the dispatcher in advance.

Another important issue addressed in this thesis is the load acceptance/rejection decisions. Under low to moderate arrival rate situations, feasibility-based acceptance/rejection decisions may be appropriate. That is, a decision maker accepts as many demands as possible in order to maximize revenue. Furthermore, if the demands have relatively tight time-windows, it may be necessary for the carriers to be more proactive. For example, the carrier may be able to predict and identify regions in which potential future demands will originate, and 'reposition' available vehicles to accept the future (unknown) demands within their time-windows. In other words, available idle vehicles move to high potential regions in anticipation of future demands. In this thesis, however, it is assumed that the time-windows are long enough Hence, the feasibility criteria would be employed for the acceptance decision under low to moderate arrival rate situations, in which most of the requested demands can be accepted and served within their time-windows.

In contrast, in over-saturated demand situations, the excessive demand not only precludes the dispatcher from accepting all the requested demands but also

provides the opportunity to select high potential demands. Since the revenue realized from loaded movements is determined by which loads are accepted, the acceptance decision should take into account the system status and the characteristics of the individual demand under consideration. In this case, feasibility-based acceptance decisions lead the system towards "saturation", whereby the system holds the maximum number of demands in the queue of vehicles. This saturated situation interferes with the efficient operation of the system with respect to the routing schedule. In other words, it becomes difficult to find reassignment opportunities to reduce the operating cost (empty movement) because of not enough room for swapping and re-sequencing. Therefore, the status of the system represented by the number of demands in the system is a possible criterion of the acceptance decision. Furthermore, fast local heuristic rules used in the hybrid solution approach can provide a good measure to characterize the potential of a requested demand in a short time. When an additional demand attribute, demand type is introduced to classify the customers according to customers' requirements, more careful exploration of the system status and demand characteristics is required.

1.4 RESEARCH OBJECTIVES

The primary objective of this thesis is to investigate key aspects of dynamic problems in fleet operation for truckload pickup-and-delivery service; namely, to analyze and formulate the dynamic fleet management problems, and to develop, analyze, and implement algorithms to solve them.

The specific goals of this research are as follows:

1. Formulate the dynamic fleet management problem under the assumption of the availability of real-time control of the fleet as well as real-time information on vehicle locations and states.

2. Develop operations research methodologies, Mixed Integer Programming (MIP) models, for static and deterministic version of the truckload pickup-and-delivery problems with time-windows; one for homogeneous demands and the other for mixed demand types with various time-window configurations.

3. Develop dynamic operational policies determining how to set local snapshot problems, how to solve the local problems using the MIP models as well as various heuristic rules, and how to apply the solutions of these problems into the dynamic operation of the fleet in a dynamic context.

4. Develop revenue management policies through acceptance/rejection decisions under various demand situations raging from a low demand arrival rate situation to an over-saturated demand situation. In addition, develop revenue management policies with multiple demand types.

5. Provide a methodology to evaluate the performance of the developed dynamic decision policies and algorithms: First, to build simulation frameworks. Second, identify evaluation benchmarks under dynamic environment.

1.5 DISSERTATION ORGANIZATION

Chapter 2 presents a review of literature and the current technologies employed in relation to this research. It begins with an introduction of Commercial Vehicle Operation (CVO) as a part of Intelligent Transportation Systems (ITS) followed by a discussion of various technologies available in the current market such as electronic trip recorders, two-way communication devices and Automatic Vehicle Location (AVL) systems. A review of the operations research literature on the topic of fleet management is also provided. The discussion begins with classical deterministic and static problems such as the Traveling Salesman Problem (TSP) and various types of Vehicle Routing Problems (VRP). Then dynamic and stochastic vehicle routing problems are discussed. Various solution approaches are presented including a discussion of stochastic programming approaches, solution approaches adopting static algorithms, and meta-heuristics. Finally, literature related to the acceptance decision problem is provided.

Chapter 3 introduces the problem definition as well as conceptual and theoretical framework for solution procedures. The general problem context is

discussed including problem statement, objectives and the time-windows properties. Then, a formal problem definition is provided. This is followed by a discussion of solution approaches, which include two MIP (Mixed Integer Programming) models for deterministic local snapshot problems and dynamic operational decision policies. Finally, a conceptual framework of a simulation, which provides the methodology for evaluating algorithm performance, is presented.

Chapter 4 introduces an idealized problem setting: a small fleet operation with moderate demand request rate. Various algorithms and heuristic strategies to solve the problem are proposed and tested. This problem is tackled by a hybrid (two phase) approach implementing local heuristic rules for constructing an initial feasible routing schedule and acceptance/rejection decision along with the MIP model for re-optimizing existing schedule.

In Chapter 5, the fleet size (and therefore, problem size) increases. Fleet size is an important factor influencing the complexity of the problem. Therefore, a large fleet with a low to moderate demand arrival rate is the next target problem setting of this thesis. To solve this large fleet problem, various partitioning strategies are developed based on a 'divide and conquer' technique.

In Chapter 6, the target problem is under an over-saturated demand situation, in which the solution algorithm execution time for a local snapshot problem is more critical than in the previous problem settings due to the relatively congested demand arrivals. In this situation, it is important to recognize the relationship between computation time and vehicle movements during this time.

Furthermore, the limited capacity of the system mainly due to the specified time-windows of the demands precludes serving all requested demands. Therefore, acceptance/rejection decision policies considering system status and individual demand characteristics are addressed along with this problem.

Chapter 7 discusses revenue management through load acceptance decision-making for two types of demand (priority and regular demands). Each customer has his/her own requirement for delivery service. Hence, a real-time truckload routing and scheduling problem associated with two classes of demand accommodating various customer requirements is targeted.

The final chapter concludes the dissertation, and provides recommendations for continuing research topics.

Chapter 2 Background Review

This chapter reviews literature on the topic of real-time information technologies and operations research literature relevant to this thesis. Commercial Vehicle Operations (CVO) applications to Intelligent Transportation Systems (ITS) are presented in Section 2.1. Section 2.2 introduces the existing models for fleet operation, which include classical problems such as Traveling Salesman Problems (TSP) and Vehicle Routing Problems (VRP). In addition, the dynamic version of VRP, which is closely related to this research, is discussed. Section 2.3 focuses on the solution approaches to these problems.

2.1 Real-time Information Technologies

2.1.1 Definition of CVO

ITS are designed to increase the safety and efficiency of surface transportation systems using advanced technologies including information processing, communications, control, and electronics. The major areas of ITS include Advanced Public Transportation Systems (APTS), Advanced Rural Transportation Systems (ARTS), Advanced Traffic Management Systems (ATMS), Advanced Traveler Information Systems (ATIS), Commercial Vehicle Operations (CVO), and Advanced Vehicle Control System (AVCS).

Most of these areas are aimed at controlling individual drivers, which prevents ITS technology from being directly implemented at the user level. This is because the required equipment costs, such as in-vehicle technologies, to the

individual users are prohibitive. Furthermore, it is impossible to fully control the activities of individual drivers. In contrast, the economic incentive from CVO application makes implementation of new technologies easier. In addition, fleet managers have the authority to control the activities of the vehicles.

The definition of CVO by the USDOT (U.S. Department of Transportation) is as follows: "*CVO include all the operations associated with moving goods and passengers via commercial vehicles over the North American highway system and the activities necessary to regulate the operations*" (USDOT, 1999). There are four main CVO applications: safety assurance, credentials administration, electronic screening and carrier operations. Carrier operations consist of three major topics: fleet and vehicle management, traveler information systems, and hazardous materials incident response. This research focuses on 'fleet and vehicle management' and more specifically on the dispatching system using real-time information.

This research assumes that real-time information is available to a dispatcher. In other words, all vehicles are equipped with an Automatic Vehicle Location (AVL) system and continuous two-way communication systems. The technologies employed in CVO will be reviewed in the following section.

2.1.2 CVO Technologies

The major technologies used for fleet management are electronic trip recorders, communication systems, AVL systems and dispatching operation

systems. A brief description of these technologies is presented in the following sections.

2.1.2.1 Electronic Trip Recorders

Electronic trip recorders are also known as onboard computers. These devices automatically monitor and record the performance of vehicles and drivers. Electronic trip recorders include a trip data display, speed and rpm audio alarms, a fuel consumption display, time of day, trip statistics and hours of service records. Furthermore, event data, discrete data and driver input can be recorded in protected areas of memory. A mobile communication system enables the driver to transfer the information to a dispatcher in real-time. As a result, a dispatcher can monitor the vehicle/driver performance historically and in real-time, this reduces the administrative workload and improves vehicle diagnostics.

Various products are available on the market today. For example, Centrodyne Inc. provides a family of "Silent 1000" trip recorders and Zepco Sales & Service Inc. products includes the ZTR 9200.

2.1.2.2 Two-way communication Systems

In this thesis, it is assumed that real-time two-way communication systems, which provide driver–to–driver and driver-to-dispatcher communications, are available. They allow a dispatcher to direct the fleet of vehicles in real-time. The available systems on the market range from mobile phones and citizen band radios to digital broadcast systems depending on cost and

sophistication. One example, Qualcomm's OmniTRACS system allows the transfer of real-time data.

2.1.2.3 Automatic Vehicle Location (AVL) System

Various alternatives are available to identify vehicle location relative to a map in real-time. The Global Positioning System (GPS), a current market leader, determines a user's worldwide location (latitude, longitude and altitude) by tracking signals from four or more satellites out of 24 satellites orbiting the Earth. The GPS can provide real-time position with accuracy within meters depending on the type of receiver and other conditions.

An announcement made by President Clinton on May 1, 2000 calling for cessation of Selective Availability (SA), which is the intentional degradation of GPS signals available to the public to prevent abuse by hostile countries, increased the accuracy of the GPS dramatically (The White House, 2000). For example, a before and after survey conducted by National Geodetic Survey (NGA, 2000) at Erlanger, Kentucky shows that 95% of the points were within a radius of 45.0 meters with SA. In contrast, 95% of the points now fall within a radius of 6.3 meters after removal of SA. Leading GPS device providers include Trimble, Novatel Inc., Magellan Corporation, THALES Navigation, and GARMIN International.

2.2 MODELING OF FLEET OPERATIONS

2.2.1 Classical Problems

This section discusses classical descriptions of fleet operations problems.

2.2.1.1 Traveling Salesman Problem (TSP)

Classical and fundamental problem in fleet operation is the Traveling Salesman Problem (TSP). The goal of the TSP is to minimize the total travel distance when a salesman, starting from his home city, is to visit each city exactly once on a pre-specified list and then return home (Hoffman, 1985). This simple problem has gained importance because not only can it be extended to many applications including vehicle routing, facility location, and machine scheduling problems but its general solution algorithm can also be used to solve other problems of its genre: combinatorial optimization. In general, the combinatorial optimization problem cannot be solved by traditional differential calculus because the decision is to find a tour (sequence) instead of determining a continuous value.

Except for the special form, for example a TSP with an upper triangular distance matrix (i.e. c_{ij} =0 if i>j), the TSP problem is a '*NP-hard*' problem. An *NP-hard* problem is extremely unlikely to have a polynomial algorithm to solve it optimally although it is not proved that the worst case exponential solution running times are unavoidable (Hoffman, 1985). The formal definition of NP-hard is as follows (NIST, 2001):

> "The complexity class of decision problems that are intrinsically harder than those that can be solved by a nondeterministic Turing machine (It is a

"parallel" Turing machine which can take many computational paths simultaneously, with the restriction that the parallel Turing machines cannot communicate) in polynomial time. When a decision version of a combinatorial optimization problem is proven to belong to the class of NP-complete problems, an optimization version is NP-hard."

The simplest approach to a TSP problem is to enumerate all possible sequences which results in a complexity of the problem equal to $O(n) = (n-1)!$ namely, the necessary computational effort grows exponentially with problem size. For example, if the computation time required for a calculation is one nano-second (10^{-9} sec), a 20-city TSP would require 3.8 years to solve. Therefore, other solution algorithms are necessary to solve this class of problems.

A mathematical Integer Programming (IP) formulation is presented before investigating the solution algorithms. Let x_{ij} be a binary variable, which indicates whether a salesman visits city j immediately after city i, and c_{ij} be the distance between the two cities (i and j). The objective function is to minimize the realized travel distance. The constraints listed below, (2) and (3), represent the flow conservation. Constraints (4) represent the subtour elimination, which prohibits the salesman from constructing subtours consisting of several subsets of cities (Wolsey, 1998). The subtour elimination constraints make the TSP hard because the constraint (4) represents a large number of constraints: 2^n-2. The mathematical formulation of the TSP is similar to the assignment problems except for the subtour elimination constraint.

$$objective \quad \min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{1}$$

$$sub\,to \quad \sum_{j=1}^{n} x_{ij} = 1 \quad for\, i = 1,...,n \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad for\, j = 1,...,n \tag{3}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1 \quad for S \subset N, 2 \leq |S| \leq n-1 \tag{4}$$

$$x_{ij} \in \{0,1\} \quad for\, i = 1,...,n,\ j = 1,...,n,\ i \neq j$$

2.2.1.2 Variations of the TSP

The TSP has evolved into various forms depending on applications; these are briefly introduced in this section.

Bottleneck TSP: The objective is to minimize the longest distance between cities rather than the sum of these distances, while a salesman travels all the given list of cities (Garfinkel, 1985).

The time dependent TSP: The costs (distance) between cities change depending on the time when a salesman travels. The objective is the same as for the original TSP (Garfinkel, 1985).

Probabilistic TSP (PTSP): The number of cities to be visited in each problem instance is a random variable where the probability that a subset of cities at an instance occurs is *P(S)* (Powell et al. 1995, Jaillet and Odoni 1988).

TSP with Time-windows (TSPTW): Each city has its own time-windows such that a salesman visits a city between the earliest and latest pickup time. This problem can be divided into two sub-problems: the TSP with hard time-windows and the TSP with soft time-windows. In the latter case, the time-windows may be violated but a penalty is then added to the objective function. Calvo (2000) proposes a heuristic algorithm for this problem based on the solution of an auxiliary problem.

2.2.1.3 Vehicle Routing Problem (VRP)

The vehicle routing problem (VRP) is a generic name given to a class of problems in which 'vehicles' visit 'customers' (Christofides, 1985). The deterministic and static version of the mathematical model that best describes certain aspects of the fleet management problem of interest falls in to this class. It seeks the efficient assignment of available vehicles to (known) demands, and the sequencing of the demands served by each vehicle, subject to various constraints.

The VRP has various forms depending on the nature of the problem such as the nature of demand, the information on the demand, the vehicle fleet, the crew, scheduling and the data requirements (Assad, 1988). A detailed classification of the problem is presented in Table 2.1.

Table 2.1 General characteristics of routing problems (Assad, 1988 p12)

Nature of Demand	Pure pickup or pure deliveries Pickups with backhaul option Single or multiple commodities Must serve all demands? Common carrier option Priorities for customers
Information on Demand	All demands known in advance? Many repeat demands Fixed frequencies for visits Uncertain demands Real-time inflow of demands
Vehicle Fleet	Homogeneous fleet or multiple vehicle types Weight and capacity restrictions Compartments Loading restrictions/equipment Vehicle type/site dependencies Vehicle type/commodity compatibility Fixed or variable fleet size Fleet based at single depot or multiple terminals
Crew Requirement	Pay structure: • length of workday • minimum and maximum on duty times • overtime option Fixed or variable number of drivers Driver start times and locations Lunch or other breaks Multiple-day trips allowed
Scheduling Requirements	Assignment of customers to day of the week Time-windows for pickup/delivery (soft, hard) Open and close times Load/unload (dwell) times
Data Requirements	Geographic database, road networks Customer addresses and locations Travel times Vehicle location information Customer credit and billing information

Solomon (1987), Christofides (1985b), and Golden and Assad (1986, 1988) provide extensive survey papers for Vehicle Routing Problems. Solomon and Desrosiers (1988) concentrate on 'VRP with Time Windows' (VRPTW), in which they discuss the single and multiple traveling salesman problems, the shortest path problem, the minimum spanning tree problem, the generic vehicle routing problem, the pickup and delivery problem, the multiperiod vehicle routing problem and the shoreline problem. Note that, the VRP, in its various forms, is a notoriously *NP-Hard* problem.

2.2.2 Stochastic/Dynamic Vehicle Routing Problems

Stochastic and static versions of the VRP called SVRP have random components in problem settings. Stewart and Golden (1983) formulate the problem as a stochastic programming problem with recourse. Berman and Simchi-Levi (1989) consider the TSP with random travel times between demands, and focus on the optimal depot. Another direction is to model a problem called probabilistic TSP, in which the demands are identified in advance only by a probability distribution. The objective is to find optimal a priori routes and to update them. This model is initiated by Jaillet (1985, 1988). Bertsimas et al. (1990) also discusses the topic.

The Dynamic VRP is the dispatching system of vehicles to satisfy a multiple demand service that evolves in a real-time (dynamic) fashion (Psaraftis, 1988). In this system, the input information changes and is updated either during the execution of the route, or during the execution of the algorithm that solves it.

This differentiates the problem from a static vehicle routing problem, which is based on known information. Furthermore, Psaraftis (1988) identifies the difference between the dynamic and the static VRP as follows: (1) Time dimension is essential; (2) Problem may be open-ended; (3) Future information may be imprecise or unknown; (4) Near-term events are more important; (5) Information update mechanisms are essential; (6) Resequencing and reassignment decisions may be warranted; (7) Faster computation times are necessary; (8) Indefinite deferment mechanisms are essential; (9) Objective function may be different; (10) Time constraints may be different; (11) Flexibility to vary vehicle fleet size is lower; and (12) Queueing considerations may become important.

Various dynamic VRP models have been proposed and are summarized below. Golden and Assad (1986) present various perspectives on vehicle routing methodologies and technologies in a practical environment. Dejax and Crainic (1987) provide in their survey paper, a review on the major research trends of empty flow and fleet management models. They also discuss the advantage of dynamic flavor models (a hierarchically integrated approach for the simultaneous management of empty and loaded freight vehicle movements). Psaraftis (1988) provides an extensive review of dynamic vehicle routing problems and identifies important issues along with static a version of VRP. Powell, Jaillet, and Odoni (1995) present a comprehensive discussion of the issues associated with dynamic network modeling problems that arise in logistics and distribution systems, and of solution approaches to these problems, including a priori optimization and on-line decision policies for stochastic routing problems. The context of this review

includes (1) the definition of specific application areas; (2) development of tractable mathematical models, with particular emphasis on optimization under uncertainty; (3) development of efficient formulation and solution algorithms; and (4) the evaluation of alternative models. Specific solution approaches are introduced in the following section.

2.3 SOLUTION APPROACHES

There are two main approaches to solve this dynamic problem: adopting of static algorithm vs. stochastic models. The latter incorporates uncertainty explicitly into the model. Uncertainty arises from various exogenous sources such as demand forecasts, external supplies of equipment and drivers, performance of the network, management of the network and errors in the data (Powell et al., 1996). The incorporation of uncertainty may make a model explosively large resulting in an intractable one. On the other hand, the former are easy to formulate and can use existing static solution algorithms. Nevertheless, they may be too sensitive to the updated information resulting in cost increase.

2.3.1 Stochastic Approaches

One of the approaches to solve this class of problems is to predict future demands for service and availability of vehicles in making assignment decisions. Before addressing specific references, it is necessary to identify the Dynamic Vehicle Allocation (DVA) or Assignment problems. DVA problems assume that a vehicle is dedicated to a single customer at a time, thus it cannot take another

demand until the current demand is completed, and it cannot construct a tour schedule explicitly.

Leading contributions to these approaches are attributed to Powell. He proposes a formulation based on the Markov decision process (1986b), algorithms for the DVA problems (1988), and a hybrid stochastic formulation of the dynamic Vehicle assignment problem (1996). Frantzeskakis and Powell (1990) use linear approximation to solve this DVA problem. Powell also presents an operational planning model (1987) and a restrict recourse strategy with random arc capacities with co-workers (Powell & Frantzeskakis, 1994). Bertsimas and Van Ryzin (1991) introduce and analyze a model for the stochastic and dynamic vehicle routing problem in the Euclidean plane with a single vehicle and extend their study (1993) to multiple vehicle problems with random on-site service time, in which they consider pure pick-ups or pure deliveries.

2.3.2 Adoption of Static Algorithm

Another line of research is based on the adoption of static algorithms. As new input occurs, static snapshot of local problems are solved repeatedly. This approach is classified into two categories depending on the time horizon to be considered (Ichoua et al. 2000).

2.3.2.1. Fast Local Operation

Fast local operations, which mainly use an insertion technique, fall into the first category. This method is easy to implement and delivers a fast

computation time. This 'fast local' operation is appropriate for a dynamic environment, where computation time is an important factor. The works of Regan, Mahmassani and Jaillet (1996a, 1996b, 1997, 1998) use this fast methodology, which explicitly recognizes the expanded set of choice dimensions available to the operator as a result of real-time information. They present and investigate various local rules for the dynamic assignment of vehicles to loads under real-time information. These computationally efficient procedures are evaluated using simulation experiments that illustrate the relatively good performance attainable through these heuristics in a stochastic dynamic environment. However, the results also suggest that one can do better through formal optimization procedures of judiciously formulated mathematical models. In other words, these approaches lose the opportunity of re-optimizing existing routes by swapping and/or reordering to improve the solution. Powell, Towns and Marar (2000) investigate the trade-off between myopic and global optimality by applying a simple load-matching algorithm repeatedly for local problems.

2.3.2.2 Sophisticated Static Problem-Solving Procedure

The second category is a sophisticated static problem-solving procedure, which involves a re-optimization of existing routes. In this category, Yang, Jaillet, and Mahmassani (2000, 2002) introduce formal optimization-based approaches, in which they introduce a formal optimization-based approach and test it against the previously developed heuristic rules (Regan, Mahmassani & Jaillet, 1996a). This approach consists of solving a mathematical programming formulation

embedded in a rolling horizon framework for the dynamic assignment and sequencing of trucks to jobs when requests for service arise on a continuous basis. The MIP formulation of the problem faced at each stage allows for the dynamic reassignment of trucks to loads, including diversion of a truck to a new load when already en-route to pick up another load, as well as for the dynamic resequencing of the order in which loads are to be served, as new loads arrive and conditions unfold. Loads have associated time windows for pick up and delivery, and the objective function includes an explicit penalty cost for not serving a particular load. The resulting off-line VRP problem is solved using a commercial optimization solver (CPLEX in this case). However, applicability of that approach is at present limited to relatively small problems, because of its computational intensiveness. This is a serious drawback from an operational standpoint in the time-constrained dynamic decision problem environment described earlier. Furthermore, it is still only a local optimization approach from the standpoint of the entire dynamic problem horizon as it cannot guarantee global optimality over the entire stream of still unknown demands, given the stochastic dynamic nature of these systems.

Powell, Snow, and Cheung develop heuristic approaches (2000) in which they introduce a mathematical formulation and two optimization-based heuristics. Comparison with one of these algorithms is presented in Chapter 5.

2.3.3 Meta-heuristics

Another solution approach is to use meta-heuristics based on a searching algorithm. The effectiveness of these methods depends on their ability to adapt to a particular realization, to avoid entrapment at a local solution, and to exploit the basic structure of the problem. The drawback of these approaches is that the solution quality is unstable. In other words, the solution quality depends on the parameter settings. As a result, new parameter settings might be required depending on the problem realization.

2.3.31 Tabu Search

The basic concept of 'tabu search' is as follows. Tabu search starts at an initial solution and allows moves to inferior neighboring solutions. This movement utilizes a list of 'tabu moves', which are unallowable moves, as they are likely to return the search to a neighborhood already investigated. Badeau et al. (1997) propose a parallel tabu search algorithm for the static version of VRP with time windows and Gendreau, Guertin, Potvin and Taillard (1999) introduce parallel tabu search heuristics for real-time vehicle routing and dispatching problems. In addition, Ichoua, Gendreau and Potivin (2000) investigate a diversion issue in the class of pick up only problems based on the tabu search algorithm. The diversion means redirecting an empty vehicle on its way to make a pick up to a newly arrived and more profitable demand.

2.3.3.2 Other Meta-heuristics

In addition to the 'tabu search' algorithm, other promising meta-heuristic algorithms are 'simulated annealing' and 'genetic algorithm'. The simulated annealing is a local searching procedure designed to avoid being entrapped at a local optimum by allowing moves to inferior neighbor solutions with a probability proportional to difference and cooling temperature. Chiang and Russell (1996) present an algorithm for VRPTW based on this meta-heuristic.

Inspired from biological evolution, in which organic methods for encoding the structure of living beings operates on chromosomes rather than on beings, the genetic algorithm was developed. Jung and Haghani (2000) propose a genetic algorithm for pickup and delivery problems with time windows.

2.4 ACCEPTANCE DECISION

Airline yield management problems, in which seat inventory control among the various fare classes has been studied for a long time, provide good insight into real-time acceptance decision-making in a dynamic routing and scheduling problem with priority demand. McGill and Van Ryzin (1999) provide a comprehensive overview on the revenue management problems. Liang (1999) formulates the yield management problem as a continuous time, stochastic dynamic programming model. Other principle problems presenting conceptual ideas for the acceptance decision problem are the various forms of the dynamic and stochastic knapsack problem (Kleywegt & Papastavrou, 2001; Papastavrou et al., 1996). Kleywegt and Papastavrou (1998) present the acceptance and

dispatching policies for a distribution problem. However, most of the research efforts in this field assume that the capacity of the system is fixed so that each accepted demand would reduce the capacity. In contrast, in the dynamic routing and scheduling problem the capability of the system to accept requested demands would be recouped as the system completes the delivery service. Furthermore, the system shows a different level of capability to accept the different types of demands due to specified time-windows. In other words, the system at a given status in which a demand is feasible, cannot accept another demand, which has a relatively short time window.

Chapter 3 Conceptual Framework

The principal focus of this chapter is to provide specific context and formal definition of the problem addressed in this thesis and to present the underlying concepts for the solution approaches. The particular dynamic truckload pickup-and-delivery problem that motivates this study is one in which customers with time-sensitive load requests call the trucking company on a continuous basis, and expect a response regarding acceptance or rejection of the request within a short time. The carrier with a fleet of trucks should serve the accepted demands by picking up the demands at their origins and delivering them to the destinations within their specified time-windows. In order to manage the dynamically requested demands, a dispatcher or decision maker needs to decide whether to accept or reject a demand upon arrival of the request, and make demand-truck assignment decisions for the accepted demands, i.e. when and which trucks serve the accepted demands within specified time-windows. The objective of the dispatcher is to maximize the overall profit while managing the service quality; the revenue earned from an accepted demand is proportional to the haul-length, and the primary operating cost is proportional to the distance traveled by a truck to serve it. To solve this problem, locally oriented hybrid operational decision policies combining fast heuristic rules with optimization-based approaches are proposed.

3.1 PROBLEM STATEMENT

3.1.1 Problem Context

The specific set of problems studied here corresponds to dynamic truckload pickup-and-delivery situations as a special form of dynamic multi-vehicle routing problems with time-windows. In these problems, once a truck is loaded at the origin of a demand, it travels to the destination and delivers the demand before picking up another demand. In this type of problem, the possibility of en-route load swap, in which two or more trucks meet at some point on the way to the destination and exchange their loads, is precluded. The dispatcher receives demand requests over a bounded geographical region dynamically. Information concerning the occurrence demands, as well as their attributes such as origin, destination and time-window (as well as demand type), is not known to the dispatcher in advance, but instead is revealed on a continuous basis as the scheduled routes and/or the solution procedures are executed. Furthermore, the dispatcher does not know beforehand the timing of when the information is updated upon being triggered by a newly requested demand. This in turn requires dynamic operation of the trucking fleet to provide highly responsive service.

Based on the insight of Regan (1997) and Powell (1996), additional assumptions are made to provide a robust definition of the dynamic fleet management problems studied in this thesis. In these problems, a trucking company with a fixed number of trucks is considered. All the trucks are empty and idle at a depot at the beginning of the day, and work on a daily basis. The dispatcher has the authority and responsibility to control all activities of the

vehicles in real-time regardless of whether the fleet is composed of company-owned vehicles, owner-operator drivers, or a mixture. Real-time information, including current location and state of the vehicles (idle, moving empty, moving loaded), is known to the dispatcher via location and telecommunication technologies. In addition, two-way communication makes it possible for the dispatcher to control the fleet in real-time.

It is assumed that the requested demands are homogeneous in the sense that they are physically substitutable, which enables the fleet of trucks to swap scheduled demands that have not been picked up. In other words, the differences among trucks in the fleet such as having a tank for petroleum, refrigerated units for food, or special equipment for hazardous material are ignored. Additionally, it is assumed that a tractor will be paired with a trailer (empty or loaded) at all times. Furthermore, it is assumed that all pick-ups and deliveries are made instantaneously.

Vehicles are assumed to travel at a constant speed according to the Euclidean travel metric. Furthermore, exogenous stochastic components including variability in truck travel time due to congestion or accidents, and in loading and unloading times spent at customer sites, are not explicitly included in the problem context. However, real-time information on the fleet of vehicles may enable the decision maker to react to this variability.

Detailed driver work regulations, specified by Federal law or local rules which limit the number of hours a driver can work at any one time are ignored.

Furthermore, the problem excludes issues such as returning drivers to their homes, suitability of the driver, required deriver skill and equipment for a load.

For this class of problem, arrival rates ranging from a moderate demand situation in which most of the demands can be served, to an over-saturated demand situation, are investigated. In the latter case, the carrier cannot serve all the requested demands. Not only does this situation force the dispatcher to reject part of the demands but it also provides the opportunity to select highly profitable demands. In addition, in a future problem specification, the customers are classified into two groups, time sensitive vs. price sensitive.

Note that the problems stated here simplify the real-world trucking problem by ignoring certain features, as stated above, but solution of these idealized problems can provide useful insights for the corresponding real-world fleet management problems.

3.1.2 Objectives and Criteria

In general static and deterministic routing and scheduling problems, in which all data are known in advance and will not change, the criterion for evaluating the quality of a solution (typically in the form of routing schedule) is relatively obvious. If all demands can be served, the standard objective is to minimize cost subject to the constraints of serving all the demands. Alternatively, some models capture the rejection costs as well as the effect of poor service in the objective function while relaxing the constraints of accepting all demands. In a dynamic environment, however, the objective of a dynamic fleet management

model depends on the particular application (Powell et al, 1995). Minimizing cost over the time horizon of interest is still one of the standard measures for many applications. However, for applications where serving all demands all the time is not guaranteed, maximizing profit capturing the rejection cost is a more realistic objective. Other objectives may include maximizing throughput and minimizing the average time in the queue of the dispatching system. Several studies have focused on the average time a demand spends in the system. For example, Gans and Van Ryzin (1999) introduce a dynamic routing and consolidation model, in which a finite number of load types arrive randomly over time to a distribution center and wait for delivery service. A vehicle with limited capacity dynamically selects a route characterized by the number of each type of load as well as by the time required to complete the route. This problem, characterized by variability of load arrivals over time as well as variability of the times required to deliver loads, tends to exhibit inevitable queuing delays. For a shipper, these delays introduce inventories waiting for service, and may cause an increase in inventory facility space. For a carrier, increasing capacity to reduce delay increases the risk of underutilizing its transportation assets. Hence, minimizing the average time the demands spend in the system is a valid objective of a dynamic problem.

In this thesis, the primary objective for fleet operation is to maximize 'profitability' while managing 'service quality' within acceptable level. Profit is equivalent to earned revenue minus required cost. Since it is assumed that the revenue is proportional to the haul-length (as well as depending on the demand type in a particular problem specification), the revenue is managed through a

load-acceptance decision. The cost is composed of fixed and variable components. The cost associated with the vehicle is a fixed cost because it is assumed that the fleet size is fixed. This assumption stems from the fact that the fleet size is determined by a long-term plan and cannot be modified in the short run. The variable operating cost is incurred through the traveled distance, including loaded and empty movements. Once a demand is accepted, the associated loaded distance is fixed. On the other hand, the empty movement required to serve an accepted demand is still variable until the demand is picked up because the dispatcher can modify the routing schedule. Therefore, one of the main responsibilities of the dispatcher is to minimize the variable costs particularly the empty movements by appropriate assignment (routing schedule) decisions. Note that, the assignment decisions have significant impact on the fleet capability of accepting demands.

The service quality, another main issue addressed in this research, is measured by 'response time'. The response time represents the time elapsed from the demand arrival time to the time when the customer receives the acceptance/rejection decision. If this time is too long compared to a customer's expectation, the dispatcher may lose the customer, particularly in this dynamic environment since it is assumed that most of the customers use telephone or Internet. Hence, the decision maker should respond to a customer within tolerable time. Furthermore, in the setting of this thesis, "waiting time" represents the time elapsed between the arrival of a demand and its eventual pickup time, i.e. the amount of time the demand spends in the queue of the dispatching system. The

proposed dynamic operational decision policies in Chapter 4 and 5 discuss the concept of "waiting time" in detail. The Dynamic Adaptive Dispatching (*DAD*) system proposed in Chapter 6 explicitly analyzes the response time. Furthermore, Chapter 7 introduces an advanced time-window structure, which captures the effect of delay.

3.1.3 Time-Windows

This section discusses the time-window constraint that represents the time sensitivity of the demand and limits the fleet capacity to accept requested demands. The time-window constraint associated with a demand represents the time interval in which the demand should be served. It can be classified into two types: hard time-windows and soft-time windows. A hard time-window constraint means that the time interval should be strictly respected. That is, the constraint should not be violated under any circumstance. In contrast, soft time-windows allow some violation, i.e., not all demands must be served within their time-windows, but when the demand is not served within its specified time-window a penalty is typically incurred. This is formulated by introducing the corresponding constraints into the objective function. The time-window in truckload problems consists of four components: earliest pickup time, latest pick time, earliest delivery time, and latest delivery time. In this thesis, since it is assumed that the vehicles move with a constant speed, once a pickup time is specified, the delivery time is consequently specified. Therefore, only two components (earliest and latest pickup times) are considered. Furthermore, it is assumed that the earliest

pickup time of a demand is identical to the arrival time of the demand request. Thus, it is impossible to pick up a demand before its earliest pickup time.

Many variations on the time-window structure exist, depending on the shippers' requirements. Figure 3.1 shows two examples, in which the solid line represents the penalty incurred, and τ^- and τ^+ denote the earliest and latest pickup times of a demand, respectively.

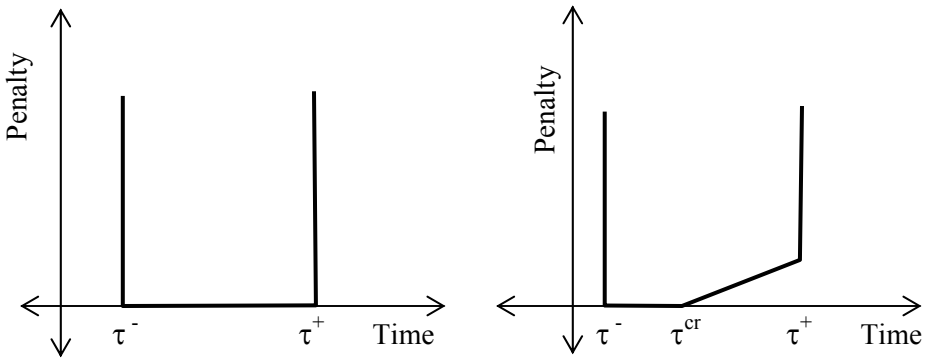


Figure 3.1 Time-Window configurations

Types I and II in the figure show the hard time-window configuration, in which the demand must be picked up within the specified time-windows $[\tau^-, \tau^+]$. Otherwise, an infinite penalty is charged to preclude serving the demand out of its specified time-window. Type I shows the case where no penalty is charged if a demand is picked up at any time within the specified time interval. In contrast, Type II introduces one more component, τ^{cr}, denoting a "critical" time such that no penalty is incurred if the demand is picked up in (τ^-, τ^{cr}). Otherwise, if the demand is picked up after the "critical time", a penalty proportional to the amount

of "over time" (from τ^{cr} to the actual pickup time) is assessed. In this way, Type II favors earlier delivery after the "critical time". In this thesis, those two types of time-windows are employed, implemented, and discussed. Furthermore, it is assumed that the width of these time-windows is relatively wide compared to the average haul-length, so that it is possible to construct a routing schedule with multiple legs.

The time-window has a significant impact on the fleet's ability to accept a demand. A demand with a wide time-window has more flexibility to be assigned to a vehicle than a narrow time-window demand. In addition, a dispatcher can hold more demands for a longer time in the queue of the dispatching system as the average width of time-windows associated with the demands in the queue increases. Therefore, under a certain demand intensity situation, as the average time-window width increases, the ability of the system to accept demands increases.

Furthermore, the time-window feature plays an important role in identifying a policy to make a delivery schedule in a fleet management problem. For example, suppose that time-windows of the requested demands are tight (the latest pickup time is close to the arrival time), the dispatcher cannot hold the demands in the queue for a long time. In other words, when a demand is requested, immediate assignment of the demand to a feasible vehicle (currently available or to be available in a short time, and close enough to the requested demand's origin) is required to satisfy the tight time-window constraints. In this case, a local snapshot problem consisting of the vehicles in the fleet and the

demands that are accepted but not-yet-picked up can be formulated as a pure assignment problem. Note that, in this formulation, the number of available vehicles should be greater than or equal to the number of demands and each vehicle can be assigned to at most one demand. This is not a *NP- hard* problem. In the extreme, the 'first come first serve' (FCFS) priority structure in conjunction with a greedy assignment strategy (a requested demand is assigned to the closest available vehicle) might be the appropriate approach to construct a feasible schedule. Furthermore, if the demand arrival rate is low, a repositioning policy of moving idle vehicles to high potential regions in anticipation of future demands might be necessary in order to reduce the risk of underutilizing the fleet of vehicles.

Conversely, wide time-windows, compared to the average haul-length of demands, provide a dispatcher with the opportunity to construct a routing schedule with multiple legs. In this case, a demand could be assigned to a vehicle that is not currently available. More specifically, the demand may be put in the service queue of the vehicle. Furthermore, a demand assigned to one vehicle could be reassigned to another vehicle or the order to serve the demand can be re-sequenced until it is eventually picked up. This reassignment opportunity can significantly improve the system efficiency although constructing the optimal routing schedule is a *NP-hard* problem even under static conditions, in which all data are known before the route is constructed and do not change afterward. Thus, constructing a routing schedule, with possible job queue for each vehicle, is taken

as the basic approach to solve this wide time-window demand problem in this thesis.

3.2 FORMAL PROBLEM DEFINITION

This section describes the notation and theoretical concepts required to describe the dynamic fleet management problem explored in this thesis. First, the notation to describe a demand is discussed, followed by a discussion of vehicle dynamics. Then, the objective function of the dynamic fleet management problem is presented.

At time *t=0*, all *K* vehicles of a truck company are empty and idle at a central depot. Transportation service (truckload pickup-and-delivery) requests dynamically arrive at a dispatcher or a decision maker of the company during a finite time horizon, [0, T] over a bounded region. At arrival instants, $\{A_i\}_{i,A_i<T} = \{A_i,\ i = 1,...,\hat{N}\}$, the demand information arrives into the system, in which A_i denotes the arrival time of the i^{th} request. Thus, $\hat{N}$ represents the total number of requested demands during the finite time horizon [0, T], i.e. $\hat{N} = \max_{A_i<T} i$. Since it is assumed that the earliest pick-up time of a demand is the arrival time, a load *i* is completely defined by the location (X and Y coordinates) of the origin ($\boldsymbol{o}_i$), destination ($\boldsymbol{d}_i$), and specified pickup time-windows (earliest and latest pickup times: τ_i^-, τ_i^+), which is denoted by a vector, $\Lambda(i) \equiv (\boldsymbol{o}_i, \boldsymbol{d}_i, \tau_i^-, \tau_i^+)$. The corresponding haul-length of load *i*, the distance between $\boldsymbol{o}_i$ and $\boldsymbol{d}_i$, is denoted as l_i.

When the carrier provides multiple levels of service, additional attributes of a demand are required. For example, the demands may be classified into two types, 'priority' and 'regular' demands, where the 'priority' demand represents a highly time-sensitive demand so that relatively short Type I time-windows are applied. In contrast, the other customers would be less sensitive to time and more sensitive to price, so 'regular', low price, service is requested. This class of demand employs relatively wide (as compared to the width of Type I) Type II time-windows, for which pickup is still allowed to occur after a critical time (τ_i^{cr}). However, a penalty is then charged depending on the excess time ($\delta_i - \tau_i^{cr})^+$ and haul-length l_i, where δ_i denotes the actual pickup time of demand i. Therefore, the vector of demand attributes is now extended to $\Lambda'(i) \equiv (\boldsymbol{o}_i, \boldsymbol{d}_i, \tau_i^-, \tau_i^+, \tau_i^{cr}, \xi_i)$, which includes the demand type (ξ_i), and the critical time (τ_i^{cr}) associated with a 'regular' demand, as shown in Figure 3.1 (Type II).

As these requests arrive, a dispatcher needs to make a series of decisions, including acceptance/rejection upon arrival of a demand and assignment of vehicles to the accepted demands. How to make these decisions along with how to apply the outcomes of these decisions to the dynamic operation of the fleet, are called a policy or strategy hereafter.

Vehicle status, location and associated schedule at time t depend on the operation policy applied. Thus, under a certain policy π, a fleet of K vehicles in a specified region at time t can be described as follows. Each vehicle has a set of attributes denoted by a vector ($I^{\pi}(k, t)$) representing current location ($\boldsymbol{o}^{k,\pi}(t)$) and

status ($s^{k,\pi}(t)$) of vehicle k at time t under policy π, where ($s^{k,\pi}(t)$) can take three values: 1 for moving loaded. 2 for moving empty (towards the next demand pick up), 3 for idle status of vehicle k. The routing schedule of truck k at time t under the last updated routing schedule determined by policy π is represented as follows: Let $q_j^{k,\pi}(t)$ denote the j^{th} scheduled load in vehicle k's job queue under policy π at time t. Then the vehicle's queue is described by a vector $q^{k,\pi}(t) = \{q_1^{k,\pi}(t), q_2^{k,\pi}(t), ..., q_{\mathrm{j}}^{k,\pi}(t)\}$, and $|q^{k,\pi}(t)|$ denotes the queue length of vehicle k and $q^{k,\pi}(t) = \phi$ represents that the vehicle k is idle. Furthermore, the overall routing schedule of the fleet at time t is denoted by $Q_t^{\pi} = \{q^{k,\pi}(t),\ k = 1,..., K\}$. $\Gamma_t^{\pi} = \{(\Gamma^{\pi}(k,\ t)),\ k=1,\ ...,\ K\}$ and Q_t^{π} fully describe the current (at time t) dynamics of the operational system under policy π.

The objective is to find a policy π, which maximizes the total profit over demands requested during the time horizon [0, T]. The policies under consideration are limited to utilize only known and certain information. The proposed policies, however, are analyzed and evaluated with probabilistic assumptions about the demands. The attributes of a demand, such as locations of origin and destination, time-windows (and demand types) follow certain probability distributions. For example, in most of the numerical experiment conducted in this study, the origin location specified in terms of X and Y coordinates, is assumed to follow a uniform distribution over the specified region. The actual locations of specified loads only become known to the dispatcher at the corresponding arrival instants. The arrival instants $\{A_i,\ i = 1,..., \hat{N}\}$ follow a Poisson process with arrival rate λ.

The objective function is defined as follows. Three variables are defined to record the outcomes of decisions under a certain policy π. First, for the acceptance/rejection decision, let D_i^π denote the decision of whether to accept or reject the requested demand *i* when a certain policy π is applied as follows:

$$D_i^\pi \equiv \begin{cases} 1 & \text{if request } i \text{ is accepted} \\ 0 & \text{if request } i \text{ is rejected} \end{cases}$$

Second, let φ_i^π denote the transportation cost required to serve load *i*. This value is based on the empty traveled distance to pick up the load. Note that, when en-route diversion is allowed, φ_i^π should be carefully recorded to consider the redirected cases. Finally, δ_i^π represents the pickup time of demand *i*. The last two variables, φ_i^π and δ_i^π are obtained from the routing schedule that a fleet of vehicles actually follow under policy π. In addition, two parameters R_i and β are defined; R_i denotes the reward from load *i*, which is proportional to the loaded distance (l_i) and price rate (r) such that R_i = r * l_i. Let β denote the transportation cost per unit distance. Then, the total profit obtained under policy π is defined as

$$V^\pi = \sum_{i}^{A_i < T} D_i^\pi (R_i - \beta(\varphi_i^\pi + l_i))$$

V^π captures the fleet's total revenue and transportation cost including traveled empty distance and loaded distance under policy π. Therefore, the overall objective is to find the optimal policy π^*, if such a policy exists, that generates maximum total profit V^*, where Π represents a set of all possible policies under consideration.

$$V^* = \max_{\pi \in \Pi} \sum_{i}^{A_i < T} D_i^{\pi} (R_i - \beta(\varphi_i^{\pi} + l_i))$$

Under situations, in which most of the requested demands can be served, assignment decisions may dominate the overall profit. Thus, applying a simple acceptance decision policy, in which all feasible demands would be accepted, along with the optimal assignment decision policy may produce similar results to the optimal policy. In other words, the number of accepted demands under the optimal policy would be near equal to the total number of requested demands ($\sum_{i}^{A_i < T} D_i^{\pi^*} \approx \hat{N}$), where $\hat{N}$ denotes the total number of requested demands during the time horizon [0, T]. In this case, the objective can be simplified to find the minimum cost policy as follows.

$$V'^* \approx \min_{\pi \in \Pi} \sum_{i}^{A_i < T} \varphi_i^{\pi}$$

When two classes of service are considered, the reward from a load (R_i) also depends on the type of the demand (ξ_i =1 regular demand, ξ_i =0 priority demand). In other words, different price rates per unit-loaded distance are applied ($r_0 >> r_1$). In addition, another cost is introduced, the lateness penalty, which depends on $(\delta_i^{\pi} - \tau_i^{cr})^+$ and the haul-length (l_i) of demand *i*. Therefore, the optimal total profit is presented as follows.

$$V''^{*} = \max_{\pi \in \Pi} \sum_{i}^{A_i < T} D_i^{\pi} (R_i - \beta(\varphi_i^{\pi} + l_i) - \gamma l_i \xi_i (\delta_i^{\pi} - \tau_i^{cr})^{+})$$

where γ denotes a scaling factor to determine the penalty contribution to the objective function.

3.3 SOLUTION APPROACHES

This section discusses the nature of decisions required of the dispatcher, including the load acceptance decision and the assignment decision (of a load to a vehicle's service queue). This leads to a discussion of solution approaches including local snapshot problem formulations, and the concept of various dynamic operation policies using these formulations.

The acceptance/rejection decision is made upon arrival of the demand, or within a short time. This decision is permanent, and cannot be reversed. Once a load is rejected, it cannot be recalled in the system. As a result, the acceptance/rejection decision governs the revenue of the carrier. On the other hand, the assignment decision of a load to a vehicle is made in the manner of a routing schedule associated with the vehicle. The routing schedule can be modified in response to updated information and/or reassignment decisions. A load assigned to a vehicle can be reassigned to another vehicle, and order of service can be re-sequenced repeatedly until the load is eventually picked up. Furthermore, it is not necessary to inform the driver of this decision until he/she is ready to travel to pick up the assigned demand. Typically, when a driver arrives at an unloading site, the next job is assigned by the dispatcher. In this way, a driver

need not be aware of these frequent modifications. However, in some cases, a driver on the way to pick up a scheduled demand may be diverted by the dispatcher to pick-up another demand. Such en-route diversion refers to redirecting of a moving, empty vehicle to another load (Regan et al., 1995, Ichoua et al., 2000). The routing schedule determines the movement of vehicles, particularly the empty movement. Thus, it is in this way that the routing schedule governs the operating cost of the fleet.

The basic scheme to solve the dynamic fleet management problem is to solve successive deterministic local snapshot problems repeatedly, as close to optimality as possible whenever new input occurs. Note that, although the optimal solution for a local problem may obtained, such optimality is only local (myopic), as there is no guarantee that a succession of locally optimal solution would be optimal in a global sense (hind sight), over the entire sequence of demands unfolding over the period of interest. Powell, Towns and Marar (2000) show that a sub-optimal solution for a particular local problem instance could even outperform the local optimal solution over time in a wide range of conditions, particularly in the presence of a large amount of uncertainty. However, they use a simple load-vehicle matching model for the local problems. In contrast, due to the short-haul property of the demands and sufficient width of the time-windows along with high enough arrival rate, it is possible to construct a routing schedule of multiple legs in this problem. Although these local vehicle routing problems are known to be *NP-Hard* problems, constructing a routing schedule for each vehicle rather than matching a vehicle to a demand may lead to better

performance in the global sense. Therefore, all the policies in this thesis seek the optimal solution for each (local) snapshot problem.

All the policies considered in this thesis are locally oriented with respect to temporal and algorithmic considerations. The time frame to be considered is local in the sense that a decision (particularly assignment decision) of a policy at time t governs the vehicle movements until the next decision is made. Furthermore, the policies are restricted to utilize only current and past, rather than forecasted information, making few or no assumptions about future demand (for acceptance/rejection decisions, it is assumed that the average demand intensity is known to the dispatcher in advance; furthermore, revenue management policies in chapter 7 consider probabilistic assumptions about demand location). The proposed policies pursue local myopic optimal routing schedule assuming no future demands rather than explicitly seek a global optimal routing schedule.

The process of updating the routing schedule is triggered by new load arrivals and reassignments of loads. Note that vehicles continue along the previously established schedule until the completion of a decision process. A new routing schedule does not become available instantaneously upon arrival of a new load or initiation of the reassignment process. The time to obtain a solution must be recognized as an integral element of the dynamic solution process. This issue is discussed in detail in Section 3.3.2 and in Chapter 6

A local snapshot problem defined with the entire fleet of available trucks and the entire set of known accepted and unserved demands may be too large to solve optimally in a sufficiently short time, notwithstanding continuing rapid

developments in both computing power and algorithmic software efficiency. Therefore, the following sub-sections present Mixed Integer Programming (MIP) formulations for the local problems, and dynamic operation policies determining how to set up the local problems and how to apply the solutions of the local problems to the dispatching system in a dynamic context. In addition, the policies regarding the acceptance/rejection decision of the dynamically requested demands are presented.

3.3.1 Local Snapshot Problem Formulations

In the dynamic decision process considered in this problem context, the objective of the local problem is to serve all known accepted demands with a given set of vehicles, at least cost without consideration of any future demand. However, the formulations presented here are more general, in that they also allow acceptance/rejection of load requests. Special cases of theses formulations are then implemented to reflect specific operational policy assumptions and dynamic decision strategies. Two Mixed Integer Programming (MIP) formulations are presented. The first considers homogenous demands, while the second considers two classes, regular and priority demands.

The first MIP model is based on Yang, Jaillet and Mahmassani's formulation (Yang et al. 1999, 2002). Since it is assumed that the vehicles move with a constant speed, time and distance are used interchangeably in this formulation. There are K vehicles (1, …, K) and N demands at time t in a local problem, where K and N may be less than the fleet size and total number of

demands in the system depending on the dynamic operation policy being applied. Some policies define a local problem with subsets of vehicles and demands. For notational simplicity, however, K is used to represent the number of vehicles in a local problem. Note that the demands that have already been served or are currently being served, at the given decision epoch, are not considered.

MIP Formulation I: Homogeneous demands

$$\min \sum_{k=1}^{K}\sum_{i=1}^{N} d_{0i}^{k} x_{k,K+i} + \sum_{i=1}^{N}(\rho l_i \, x_{K+i,K+i} + \sum_{j=\{1,,N\}, j\neq i} d_{ij} x_{K+i,K+j})$$

subj to

$$\sum_{v=1}^{K+N} x_{uv} = 1 \qquad \forall\, u = 1, ..., K+N \qquad (1)$$

$$\sum_{v=1}^{K+N} x_{vu} = 1 \qquad \forall\, u = 1, ..., K+N \qquad (2)$$

$$x_{uv} = 0,1 \qquad \forall\, u, v = 1, ..., K+N \qquad (3)$$

$$-\sum_{k=1}^{K}(d_{0i}^{k} + v^{k}) x_{k,K+i} + \delta_i \geq 0 \qquad \forall\, i = 1, ..., N \qquad (4)$$

$$(l_i + d_{ij}) x_{K+i,K+i} - M x_{K+i,K+j} - \delta_i + \delta_j \geq -M + l_i + d_{ij} \quad \forall\, i, j = 1, ..., N \qquad (5)$$

$$\tau_i^{-} \leq \delta_i \leq \tau_i^{+} \qquad \forall\, i = 1, ..., N \qquad (6)$$

This formulation corresponds to a routing and scheduling problem with time-window constraints. The objective function is to find the least-cost set of cycles that involve all the nodes of (*1, ..., K, K+1, ..., K+N*) where node k (k = 1, …, K) represents vehicle k and node $K+i$ (i = 1, …, N) corresponds to demand i. The binary decision variable x_{uv} (u, v = *1, ..., K+N*) indicates whether arc (u, v)

is selected in one of the cycles; $x_{k,K+i}$ (k = 1, …, K; i = 1, …, N) indicates whether vehicle k serves demand i first, $x_{K+i,K+j}$ (k = 1, …, K; i,j = 1, …, N) indicates if demand j is served immediately after demand i, $x_{K+i,K+i}$=1 (k = 1, …, K) means that demand i is rejected, and $x_{k,k} = 1$ represents that vehicle k will be idle. Furthermore, the continuous variable δ_i represents the pickup time of demand i.

The input parameters are extracted from the information collected at decision epoch t. The time that vehicle k will become available is denoted by ν^k, given the vehicle's status at a decision epoch. A similar updating procedure is applied to d_{0i}^k, which denotes the distance from vehicle k's updated location to demand i's origin. These updating procedure must also take into account the computation time required to solve the local snapshot problem. This issue is discussed in detail in Section 3.3.2.2. The parameters τ_i^-, τ_i^+ represent the earliest and latest pickup times of demand i, respectively; d_{ij} represents the distance from demand i's destination ($\boldsymbol{d_i}$) to demand j's origin ($\boldsymbol{o_j}$), and l_i represents the loaded distance of demand i. In addition, M is a constant large enough to let constraint (5) be nonrestrictive when $x_{K+i,K+j} = 0$.

The acceptance/rejection decision in this formulation depends on the parameter ρ. If a demand i is rejected ($x_{K+i,K+i}$ =1), a penalty proportional to the haul-length (ρ * l_i) is incurred. In a dynamic environment, it may not be acceptable to reject demands that were accepted in previous decision epochs. To preclude this occurrence, $x_{K+i,K+i}$ is pre-specified as 0 for all existing demands other than the newly arrived demand. Alternatively, if the local problem is defined

for situations where all the demands must be served (for example, as part of reassignment procedure) ρ could be specified as a large constant to preclude any rejection.

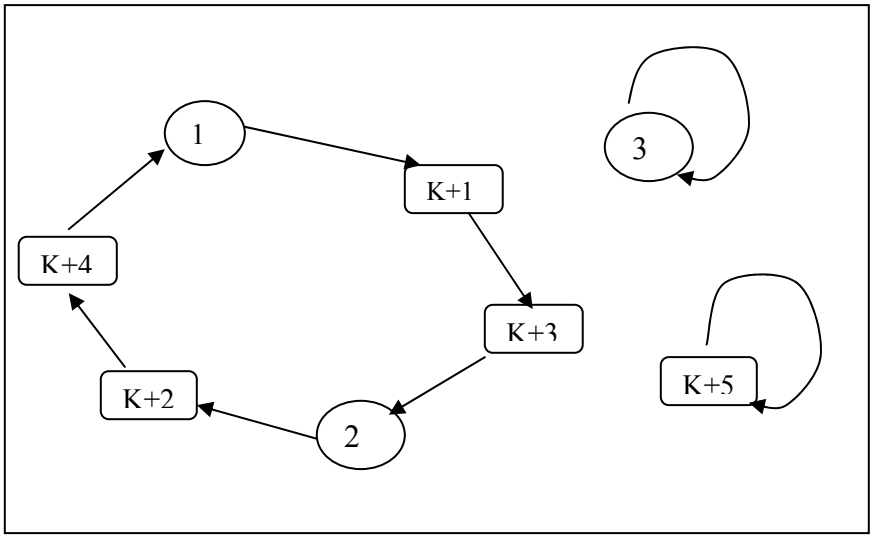


Figure 3.2 Example of a solution of MIP

The solution of this MIP formulation produces cycles, which include at least one vehicle node due to the constraints (4) and (5), except for the isolated (rejected) demands, for which $x_{K+i,K+i}$ =1. A cycle is translated into vehicle schedules that serve the demands. For example, in Figure 3.2, a cycle consisting of 1, K+1, K+3, 2, K+2, K+4,1 is interpreted such that vehicle 1 serves demands 1 and 3 sequentially, and vehicle 2 serves demands 2 and 4. A cycle with one vehicle node (3) indicates that this vehicle will be idle, and a cycle with one demand node (K+5) indicates that the demand is rejected. As the objective function does not involve the pick up time of demand i, δ_i can take any value

within the boundary of the constraints. Thus, when constructing a routing schedule δ_i is specified as the earliest possible pickup time following the order of service.

MIP Formulation II: Two demand classes

$$Min \sum_{k=1}^{K}\sum_{i=1}^{N} d_{0i}^{k} x_{k,K+i} + \sum_{i=1}^{N}(\rho l_i \, x_{K+i,K+i} + \sum_{j=\{1,,N\}, j\neq i} d_{ij} x_{K+i,K+j}) + \gamma \sum_{i=1}^{N} l_i \, \omega_i$$

subj to

$$\sum_{v=1}^{K+N} x_{uv} = 1 \qquad \forall\, u = 1, ..., K+N \qquad (1)$$

$$\sum_{v=1}^{K+N} x_{vu} = 1 \qquad \forall\, u = 1, ..., K+N \qquad (2)$$

$$-\sum_{k=1}^{K}(d_{0i}^{k} + v^{k}) x_{k,K+i} + \delta_i \geq 0 \qquad \forall\, i = 1, ..., N \qquad (3)$$

$$(l_i + d_{ij}) x_{K+i,K+i} - M x_{K+i,K+j} - \delta_i + \delta_j \geq -M + l_i + d_{ij} \qquad \forall\, i, j = 1, ..., N \qquad (4)$$

$$B + \omega_i - \delta_i \geq B\,\xi_i - \tau_i^{cr} \qquad \forall\, i = 1, ..., N \qquad (5)$$

$$\omega_i \geq 0 \qquad \forall\, i = 1, ..., N \qquad (6)$$

$$x_{uv} \in \{0,1\} \qquad \forall\, u, v = 1, ..., K+N \qquad (7)$$

$$\tau_i^{-} \leq \delta_i \leq \tau_i^{+} \qquad \forall\, i = 1, ..., N \qquad (8)$$

The second MIP formulation considers two classes of demand. In this formulation, *B* is also specified as a constant large enough to release constraint (5) when demand *i* is a priority demand ($\xi_i = 0$). A new variable ω_i is introduced to represent the overtime, the elapsed time from the critical time (τ_i^{cr}) to the pickup time (δ_i). Due to constraints (5) and (6) as well as the objective function, ω_i takes

a positive value only if a regular demand i (ξ_i = 1) is served after the corresponding critical time ($\delta_i \geq \tau_i^{cr}$). Otherwise, the value of ω_i is zero. The penalty is proportional to the overtime ω_i as well as haul length of demand i. γ denotes a scaling factor to determine the penalty charged to the objective function.

3.3.2 Dynamic Operation Policies

3.3.2.1 Hybrid Dynamic Decision policy

The MIP formulations presented in Section 3.3.1 construct a routing schedule by solving a local problem. Even though the schedule from the optimal solution of a local problem does not necessarily achieve global (hindsight, after the fact) optimality, it can produce a local optimal schedule at each decision moment. However, the computation time to solve the MIP model is a major impediment to direct implementation of the model. The dispatcher must respond to the customer within a short time, at least with regard to acceptance or rejection of the load, which is typically ensured by a feasible assignment. Therefore, time limitation is a primary consideration and significant hurdle for the execution of optimization procedures, especially with regard to acceptance decisions. However, once a load is accepted, the manner in which it is served, namely the routing and scheduling of a truck to serve it, can be modified by reassignment as warranted by unfolding conditions, so long as the associated time-windows are respected. Nonetheless, it is still difficult to solve the local problem optimally (particularly when it involves a large size fleet of trucks) due to the complexity of the problem and unknown timing of the next demand arrival. Therefore, various

algorithms, such as those discussed in Section 3.3.2.3 and Section 3.3.2.4 are developed to solve the local problem as close to optimality as possible.

An alternative heuristic approach to solve the local problem, based on Regan, Mahmassani and Jaillet's work (1996a), is to find the minimum cost sequence of demands for each truck including the newly requested demand whenever a request is received, then compare these costs across the available fleet to identify the feasible vehicle-load assignment with the minimum additional cost. For this procedure, the MIP formulations presented in the above section can be applied to a single vehicle and its associated demands. Alternatively, it is possible to investigate all possible job-sequences for each vehicle with an enumeration method. If there is no feasible vehicle in the fleet, the demand is rejected outright. These heuristic approaches can find 'good' solutions in relatively short computing time, but with no guarantee of obtaining the local optimal solution. In other words, there remains an opportunity to obtain a better solution at each decision instance by solving the associated local problem with the MIP model.

In a dynamic environment, the above two approaches have pros and cons. The basic idea of combining the above two approaches is as follows: the efficient and quick heuristic rules are used for a feasibility check and to construct an initial routing schedule upon demand arrival, while the optimization-based procedures implementing the MIP models over the entire fleet (or a subset) of vehicles are used for improving the system efficiency before the next demand comes in. The specific procedures and detailed logic of this hybrid approach are discussed in the following chapters. Note that one of the advantages of this combined hybrid

approach is that the schedule resulting from the fast heuristic procedure provides an upper bound for the objective value and an initial feasible solution for the optimization-based approach.

3.3.2.2 Local Problem Setting

When a local snapshot problem is set for the proposed algorithm, the computation time to solve the local problem needs to be taken into consideration due to the vehicle movement that will occur before completion of the routing decision process execution. When a local snapshot problem is set on the basis of the current state of the vehicles and the known but not served demands, it is important to avoid conflicts between the routing schedule obtained in the solution and the updated locations and status of the vehicles.

For example, suppose that a vehicle is scheduled to serve load A and load B sequentially, as illustrated in Figure 3.3, where the dotted lines represent empty movements and solid lines represent loaded movements. When the computation finishes at $t+\Delta t$, the solution of this assignment may direct the vehicle to pick up load B first instead of load A, even though the vehicle has already picked up load A. To avoid this type of conflict, the local snapshot problem at time t must be carefully defined to exclude load A from the local problem. Therefore, only a subset of demands, rather than the entire set of demands in a vehicle's queue, is typically selected as candidate demands for the local problem. Not only is it possible to choose a subset of demands, but a reassignment procedure aimed at improving system efficiency may take into account a subset of vehicles and their

associated demands - particularly in problems with a large fleet size, due to the complexity of the problem. Note that, Δt in Figure 3.3 is not predictable in advance.

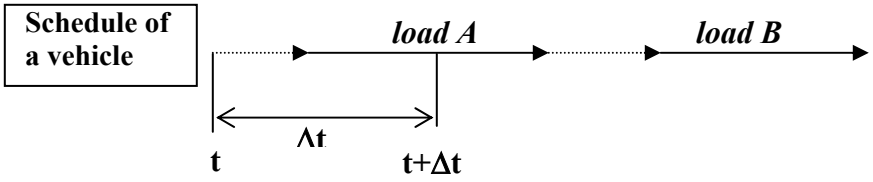
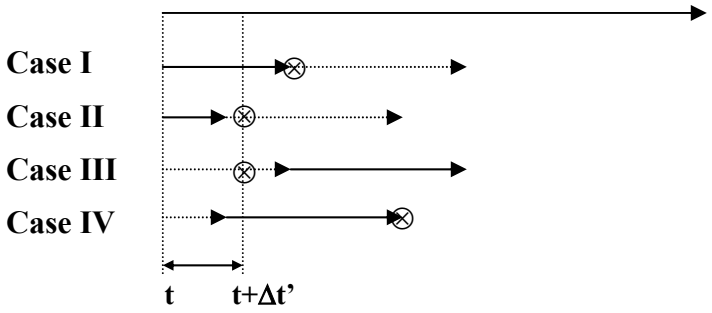


Figure 3.3: Relationship between assignment execution time and vehicle movement

When a local problem is defined, it uses updated vehicle locations and status information instead of the current information. For instance, Figure 3.4 represents various situations of a vehicle and its schedule. The computation starts at time *t,* and must be completed within $\Delta t'$, the predefined maximum allowable computation time. The solid lines represent the loaded movements and the dotted lines represent empty movements to pickup a load.

Figure 3.4 shows updated locations in various situations. If the status of a vehicle at $t+\Delta t'$ is loaded, the location of the vehicle should be updated to the destination of the load, which is the updated location at the time indicated as ⊗ (cases I and IV). If the status at $t+\Delta t'$, is empty, the input location for a local problem is the updated location at $t+\Delta t'$ as in cases II and III. If a vehicle is idle, its current location will be used in specifying the local problem. In addition, the time at which a vehicle will become available should be modified according to the updated information. The value of the allowed computation time will typically

be set to reflect the characteristics of the problem at hand, the available hardware and computational resources, as well as tolerable customer response times.



⊗: Updated location of a vehicle
Δt': Predefined maximum computation time

Figure 3.4 Computation time and local problem setting

3.3.2.3 Partitioning Strategy

Problem size, defined by the number of vehicles and the number of demands, is an important factor that critically affects the computation time for the optimization-based solution approach. The required computation effort increases exponentially with the problem size. Therefore, a local problem instance at a given decision epoch may be too large to solve optimally within an acceptable time frame. In order to overcome this obstacle, various partitioning strategies are developed based on the 'divide and conquer' technique.

The partitioning strategies are designed to select high-potential candidate vehicles and their associated demands to set a local snapshot sub-problem. This sub-problem is solved optimally. Typically, these partitioning strategies are employed for the reassignment stage of the hybrid solution approach. Furthermore, the partitioning strategies modify the part of the given schedule determined at the previous decision epoch rather than regenerating the optimal routing schedule. Detailed discussion is provided in Chapter 5.

3.3.2.4 Dynamic Adaptive Dispatching (DAD)

Another source of difficulty for the dynamic fleet management problem with respect to the proposed local optimization approaches is the unexpected inter-arrival time between consecutive demand arrivals. Typically, the solution for a local problem needs to be obtained before the next demand arrives. One way to satisfy this dynamic operational condition is to restrict the allowable computation time for solving a local problem despite problem complexity. However, imposing an arbitrary pre-determined computation time for a local problem may waste the computational resources of the system. Furthermore, when multiple customers call the delivery service almost simultaneously, it may be practically impossible to solve the local problems within the gap between successive demand arrivals.

A dynamic adaptive dispatching strategy is developed to better utilize the available computation resources and to satisfy the requirements of this dynamic operational environment. This strategy is a variation of the hybrid solution approach, in that various assignment decision strategies are adaptively applied

depending on the system state. A load arrival triggers a quick heuristic solution procedure to make the acceptance decision and the initial schedule. Then, various reassignment procedures are implemented repeatedly (along with a partitioning strategy and a process to control local snapshot problem size) until the next demand arrival. The strategy takes into account the tolerable response time concept. Chapter 6 presents a detailed discussion of this strategy.

3.3.2.5 Acceptance/Rejection Decision

The criteria for a load acceptance/rejection decision, which should be made in near real-time or at least in a very short time in order to respond to the customer, should take into account various system features. Under a low to moderate demand arrival situation, when a carrier can accept and serve most of the requested demands, the dispatcher typically tries to accept as many demands as possible in order to maximize the overall revenue and to avoid the risk of underutilizing the transportation resources. In this case, a possible criterion for the acceptance decision is the feasibility of the requested demand, which is assured by constructing a feasible routing schedule for the fleet. The hybrid policy presented in Section 3.3.2.1 provides a feasible routing schedule in a relatively short time with the fast heuristic rules. However, this feasibility check process is not based on full assessment, which should explore all possible routing schedules of the fleet.

Under over-saturated demand situations, i.e. when the demand for service exceeds the system's average capacity, the dispatcher cannot accept all requested

demands due to the time-window constraints of the demands. In other words, the number of demands the dispatcher can accept and hold in the queue of the system is limited. This limitation is called 'holding capacity', hereafter defined as the maximum number of demands waiting for service in the system. On the other hand, this situation provides the opportunity to select highly profitable demands without the risk of underutilizing transportation assets. Furthermore, it is not necessary to hold as many demands as possible in the queue of the system in order to avoid the risk. In other words, rejecting some feasible demands does not directly result in underutilizing the transportation resources. In conjunction with this condition, the number of demands in the system at a decision instant has significant impact on the efficiency of the reassignment process. When the 'holding capacity' of the system is completely filled up with the accepted demands, it is difficult to find reassignment opportunities to improve system efficiency because there is not enough room for the swapping and re-sequencing of existing demands. Therefore, an acceptance decision policy that controls the total number of demands in the system is proposed. Furthermore, the fast local heuristic rule used in the first phase of the hybrid approach can provide a good measure to characterize the potential of a demand in a short time. Thus, an acceptance decision policy considering these features is proposed. Such strategies are discussed in detail in Chapter 6.

The addition of priority/regular demand types introduces another level of complexity to the study of acceptance decision policies. Even under an identical system state, the feasibility of a regular demand is significantly different from the

feasibility of a priority demand; this is mainly due to the difference in time-window widths. In other words, the system at a given state in which a regular demand is feasible, cannot accept a priority demand. Of course, the feasibility of a demand also depends on other various features of the system. The basic idea of the acceptance decision in this scenario is to control the acceptance decision of the regular demands while accepting as many priority demands as possible since the priority demands generate greater revenue than the regular demands. In addition, accepting a regular demand may preclude the system from accepting a more profitable future priority demand. Hence, while a requested regular demand may be feasible, the acceptance decision seeks to maintain the ability of the system to accept future priority demands. Detailed discussion of the acceptance decision policy for priority and regular demands is provided in Chapter 7 along with experimental results.

3.4 EVALUATION METHODOLOGIES

The methodology by which to evaluate the solution quality of a proposed dynamic decision policy is a fundamental question in dynamic fleet management problems. For a problem, which is static and has a single well-defined objective function in spite of the difficulty of obtaining the optimal solution, comparing the objective function values of proposed solution strategies provides a fair evaluation (Powell et al., 1995). In the dynamic fleet management problem, however, the demands are typically requested under probabilistic assumptions. Thus, it is difficult to directly compare proposed strategies. Furthermore, even for a fixed

given demand stream, comparison with the global optimal solution (hindsight solution) may not be applicable since it is computationally extremely difficult to obtain the hindsight optimal solution for a reasonably sized dynamic problem. Possible methods for evaluating the strategies include deriving upper and lower bounds for the solution, testing the strategies in the worst possible scenario, and simulation experiments. This study recommends the use of a simulation framework (test bed) for the evaluation of proposed policies in comparison with benchmark policies.

3.4.1 Simulation Framework

Important features of the simulation framework required to evaluate dynamic decision policies are presented in this section. In general, a discrete-event simulation framework is employed to evaluate the performance of the proposed algorithms in this study. The fundamental events include demand arrival, pick up, and delivery as well as the end of the simulation. Corresponding to these events, the simulation defines the system state at each point. In other words, the current state of the system is updated and the current value of the simulated time (simulation clock) advances from the previous event to the next-event until the end of the simulation. Since inactive period times are skipped, the result of the system can be obtained in a relatively short time compared to real time (wall-clock time). In addition to these basic events, some policies explicitly take into account the end of the real (wall-clock) computation time of a solution procedure as one of the simulation events that update the system state. Hence, the

simulation time cannot "get ahead" of the real-world time. In other words, in order to simulate the system performance for a certain time, the simulation should be run for almost the same amount of time.

3.4.2 Benchmark

The evaluation of a proposed solution algorithm for static vehicle routing problems is relatively simple. For example, applying a solution algorithm with the benchmark problems, such as Solomon's VRPTW (Vehicle Routing Problems with Time-Windows, http://w.cba.neu.edu/~msolomon/problems.htm), for which the optimal solutions or the best-known solutions are published, demonstrates the efficiency of the algorithm. Regarding dynamic routing and scheduling problems, however, there is no benchmark problem that has been agreed upon. Furthermore, detailed specifications of the problem have a significant impact on the performance of a policy. In other words, it is difficult to develop a robust algorithm, in which the designed algorithm performs well under various conditions and demand patterns. Therefore, two recently published algorithms, which are implemented in similar problem settings, are selected.

The simple heuristic approach (Regan et al., 1996a) presented in Section 3.3.2.1 is the first benchmark algorithm. This algorithm is implemented to various problem settings. Furthermore, Powell, Snow, & Cheung (2000) published algorithms for the truckload routing and scheduling problem. One of them is the *RAPID-SL* (Resource Allocation procedure for the Integrated Dynamic

Assignment Problem-Single Label). This algorithm is implemented for dynamic fleet problems specified in Chapter 5 and the comparative results are reported.

3.5 SUMMARY

This chapter presented the conceptual and theoretical framework to define, solve and evaluate the dynamic fleet management problem. The problem is formally defined. The basic scheme to solve the problem and a brief description of the proposed policies are presented. Finally, the evaluation methodology is discussed.

The following chapters provide detailed discussion of the decision policies (strategies) and numerical results corresponding to the proposed problem specifications. First, a small problem with 10 trucks is introduced in Chapter 4. This assumes a demand situation in which most of the requested demands can be served with the proposed policies. The hybrid strategy presented in section 3.3.2.1 is applied and its performance is reported with numerical results. The relation between local problem size (number of demands involved with the local problem) and computation time to solve the local problem is analyzed as well as the policy to control the local problem size. Chapter 5 analyzes a large fleet problem with low to moderate demand arrival rates. In this chapter, partitioning strategies, which select high potential vehicles as candidate vehicles for the reassignment stage, are discussed. Chapter 6 assumes over-saturated demand situations, for which an intelligent acceptance decision policy rather than simply accepting all feasible demands is required. Furthermore, due to the short inter-arrival times

between consecutive requests, the local problem is carefully defined to avoid the conflict discussed in Section 3.3.2.2. Therefore, the *DAD* strategy is introduced and applied. In addition, a filtering strategy, controlling the total number of demands in the system, is presented. Finally, Chapter 7 introduces two types of demands and presents revenue management policies through an acceptance decision.

Chapter 4 Analysis on Small Fleet Problem

4.1 INTRODUCTION

Chapter 4 introduces an idealized problem setting with small fleet (10 trucks), under moderate demand arrival, in which most of the requested demands can be served with the available fleet with the proposed dispatching algorithms. With respect to a vehicle, the average inter-arrival time between calls for service is about 150% of the average haul-length (travel time). Because serving a load also entails empty moves by the vehicle, the resulting congestion level in this system is quite high, but it is still possible to serve most of the requested demands considering the associated time-windows of the demands and the geographical area where they are generated. In order to provide enough flexibility to construct routing schedules with multiple legs, the type I time-window in Section 3.1.3 is applied with an average width that is about 4 times the average haul-length.

The objective of this stylized problem is to find the policy, which minimize the overall operating cost (empty movements) while accepting all demands requested during the time horizon (T) as discussed in Section 3.2.

$$V'^{*} \approx \min_{\pi \in \Pi} \sum_{i}^{A_i < T} \varphi_i^{\pi}$$

Furthermore, the time horizon (T) is specified as a long enough value rather than modeling daily-based operation, in order to obtain the steady state performance measure. Therefore, average empty distance required to serve a

demand, $\sum_{i}^{A_i<T} \varphi_i^{\pi} \Big/ \hat{N}$ where $\hat{N}$ represents the total number of requested demands during the finite time horizon [0, T], is a main measure to evaluate the proposed policies.

The two-phases hybrid dynamic decision policies presented in Chapter 3 are applied to solve the problem, which involves two operational procedures controlling the local snapshot problem size in order to control the computing time of the optimization-based reassignment procedure. In this small fleet problem, the number of vehicles evolved with a local snapshot problem in the reassignment phase is fixed (10 vehicles), so only the number of candidate demands for each local snapshot problem is controlled. Simulation experiments are conducted to evaluate the performance of these strategies under alternative specifications and parameter values. The simulation results highlight the tradeoff between computation effort and solution quality.

4.2 DYNAMIC OPERATION POLICIES

4.2.1 Hybrid Dynamic Decision Policy

The logic for combining the local heuristic and the optimization-based approach, articulated in the previous chapter, arises naturally in light of the dynamic decision problem context requirements, and the respective features and characteristics of the two approaches. The basic mechanism for effectively combining the approaches is through the reassignment process, after a load has been accepted and assigned to a vehicle by the heuristic approach, and before a

new load is received. Therefore, the assignment of loads to vehicles can be viewed in two phases.

In the first phase, called initial assignment, the incoming load is assigned to a vehicle using the heuristic rule, as described in Chapter 3 if it is feasible. The acceptance criterion in this problem is feasibility of the demand. In other words, all feasible demands in the initial assignment phase are accepted. The initial assignment procedure examines feasibility and additional cost due to the new demand. The vehicle with the least additional cost is selected, and a new schedule for that vehicle is made. This additional cost can be a negative value as shown in Figure 4.1, where solid lines represent the loaded movements and dotted lines show empty movements. As can be seen, as a new demand is inserted, the total empty distance to serve demands diminishes.

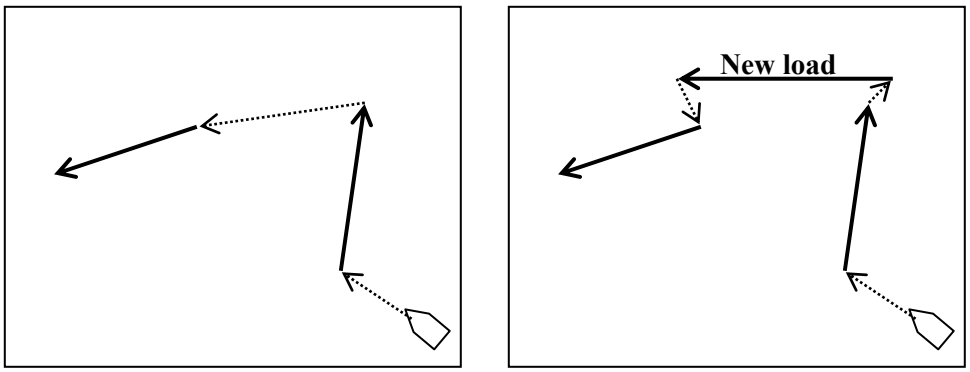


Figure 4.1 Example of a negative additional cost

As mentioned in Chapter 3, there are two methods to implement the heuristic rule. In the first, the MIP formulation in Section 3.3.1 can be applied repeatedly such that a local snapshot problem involves one truck and its

associated demands and the newly requested demand. This procedure is applied to the entire fleet of vehicles. The other method investigates all possible job-sequences (*(m+1)!*) for each truck, where *m* represents the number of demands not picked up yet in the queue of a vehicle and the additional one corresponds to the new demand. In the problem setting here with 10 vehicles with moderate demand, the latter method is applied, due to the simplicity of the method. The computing time of this method, however, increases dramatically with the queue length of each vehicle. Therefore, the former method might be preferable when the queue of the waiting jobs for a given truck is long. This initial assignment requires relatively short computation time to execute, which is necessary because the operator should respond to the customer's demand in a short time. To improve the performance of the strategy, en-route diversion is allowed in the initial assignment stage, whereby trucks on the way to pick-up some demand could divert to pick up the other demand if profitable. Since the en-route diversion decision should be made in a short time to avoid the conflict illustrated in Figure 3.3, only the initial assignment allows en-route diversion.

After initial assignment, the operator may have enough time to reconsider the vehicles' routes and schedules before another new load request comes in, though it is not guaranteed with probability one due to the nature of Poisson arrival process. This constitutes the second phase, during which reassignment is carried out using the MIP formulation I presented in Chapter 3 involving the entire fleet of vehicles and selected candidate demands. Note that to avoid the

conflict shown in Figure 3.3, the local problem for the reassignment phase does not include the first scheduled demand of each vehicle in the fleet.

Nonetheless, it is still important to control the execution time of the optimization solution procedure in the second phase, in ways that do not disrupt the system's operation, while improving overall operations by generating optimized solutions before the next decision instance. Two techniques are proposed and tested for this purpose, as described in the next sections.

4.2.1 Dynamic Control of the Local Problem Size

Dynamic control of the local problem size in terms of the number of candidate demands is proposed. The number of decision variables in the MIP formulation greatly affects the computation time to find an optimal solution. In this problem setting, the number of loads that are candidate for reassignment is controlled to a manageable level (itself dependent on the fleet size, available computation resources, and the frequency of load requests, among other considerations). The number of demands that are candidate for reassignment is controlled by a 'time cut' criterion. Given a current set of loads assigned and scheduled for each vehicle, the first demand in each vehicle's schedule will not be a candidate for reassignment. Only loads whose future scheduled pickup times are later than the predefined time cut criterion, are selected as candidates for reassignment. For example, consider the situation depicted in Figure 4.2, in which there are three vehicles and 10 demands at the given decision point. Seven of those are candidates for reassignment under the time criterion shown in the figure.

It is expected that the greater the number of candidates considered for reassignment, the better the resulting truck routing schedule. However, computing time is not linearly proportional to the number of candidates, but increases exponentially with that number as shown in the following simulation experiment. Hence the maximum number of re-assignment candidates (***max_cand***) is limited. In identifying the set of candidates for a given local snapshot problem, the cut-off time criterion is increased gradually from the current time until the number of candidate demands decreases to the predefined value of ***max_cand***. When the total number of demands is less than ***max_cand***, all demands except for the first demand in the vehicle queues are considered for reassignment.

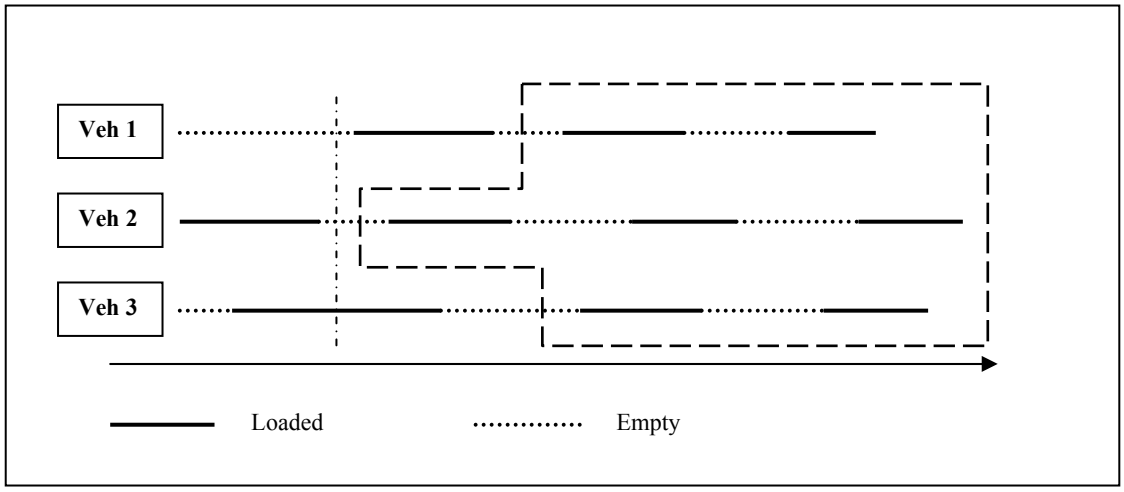


Figure 4.2 Selection of candidates for reassignment by time criterion

The rationale supporting the 'time cut' procedure, which may remove the earlier loads in the queue, is that these demands tend to have less freedom to be reassigned. That is, these demands' latest pickup times (τ_i^+) are likely to be

tighter than those of the loads at the end of a queue. In addition, this proedure helps to avoid the conflict illustrated in Figure 3.3.

4.2.2 Merge Close Demands

In an effort to further reduce the number of decision variables in the optimization stage, this procedure seeks to take advantage of the initial schedule of trucks determined as a result of applying the heuristic rules in the first phase. If two consecutively scheduled loads to be served by the same vehicle are very close spatially, as a result of the initial assignment, this order would not be likely to change in the second phase. Therefore, the gap between two consecutive demands (distance from the destination of $q_i^{k,\pi}(t)$ to the origin of the $q_{i+1}^{k,\pi}(t)$) is examined after a new requested load is assigned. If the gap is smaller than a pre-defined threshold, the two demands are merged and represented as one demand. After reassignment, the merged demands are converted back to the original two demands.

This procedure must be applied with particular caution. First, the loaded moving distance of the merged demand includes the empty distance between the two loads in addition to the two loaded distances. Secondly, the time window of the merged demand should similarly be re-calculated, taking into consideration the time windows of the two demands and the empty distance between them.

The combined approach, with the above size-reduction procedure, is tested using simulation experiments, presented in the next section.

4.3 SIMULATION FRAMEWORK

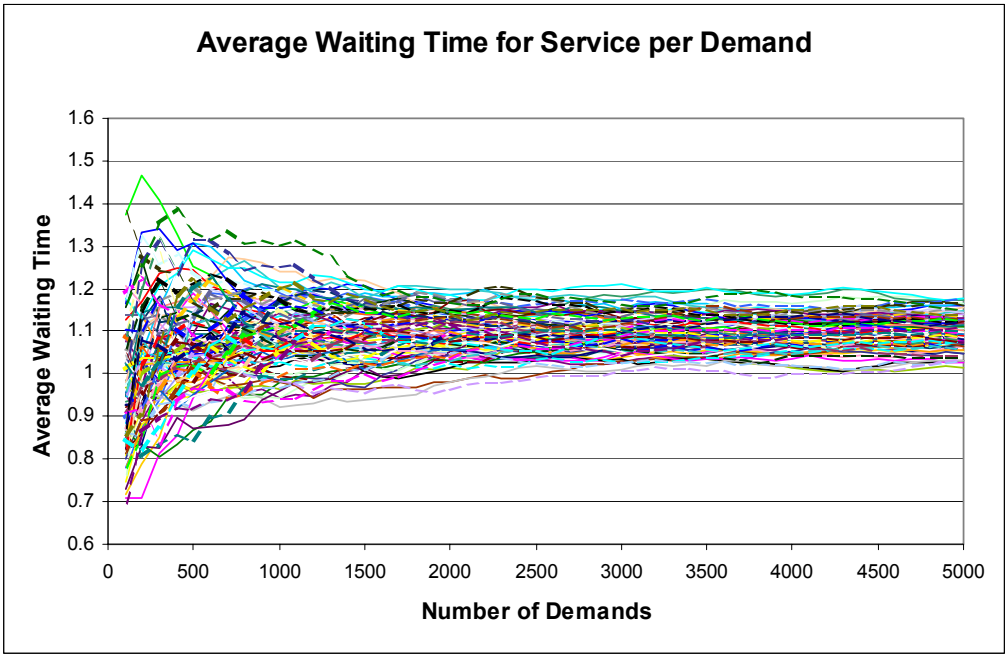
Truckload carrier operations are considered over a given 1 by 1 square unit geographical area. Calls for truckload pickup-and-delivery arise over this area and over time. A truck travels to the origin of a load, picks up the load, moves it to the destination, and then goes to the origin of the next load in the queue with constant speed 1 unit distance per unit time.

Requests are randomly generated according to a time Poisson process. It is assumed that the earliest pickup time is the calling time of the customer, and the latest pick up time is two time units thereafter. Origins and destinations of demands are uniformly distributed over the region, so the average loaded distance for a request is around 0.52 units because vehicle movement follows the Euclidean travel metric. This value limits the maximum arrival rate of demands so as not to result in an infinite queue.

All the simulation programs throughout this thesis are developed using the C computer language and the MIP models are solved using the commercial solver CPLEX 7.1. In addition, all the simulation experiments are performed on a Windows NT workstation, 300MHz CPU.

The primary performance measure used to evaluate the various policies in this chapter is the average traveled empty distance to serve a load ($\sum_{i}^{A_i<T} \varphi_i^{\pi} \Big/ \hat{N}$). This distance is the primary determinant of the cost to serve a particular demand. An additional implicit measure is the waiting time, taken here as the time elapsed between the customer's call for service to the time that the truck actually picks up the load ($\delta_i^{\pi} - A_i$). Furthermore, the computational time is also examined.

In this chapter, the performance of various solution approaches is evaluated at steady state of the system rather than assuming daily-based operation of the fleet. In order to obtain statistically meaningful results at steady state, the time horizon (T) and the number of experiments (iterations) should be determined. The first step is to check the point at which the system approaches a steady state. Figure 4.3 depicts the cumulative average of the respective performance measures as a function of the number of generated demands; each line in this figure corresponds to a stream of demands generated from a given random number seed. It appears from the figure that T generating 2000 loads ($\hat{N} = 2000$) is sufficient to ensure that the system reaches steady state. The second convergence criterion is used to determine the point at which sufficient number of iterations (each with a different seed) have been run to ensure that the performance measures aggregated over a certain number of iterations have converged to an acceptable range. In this case, 20 iterations are sufficient for this purpose.



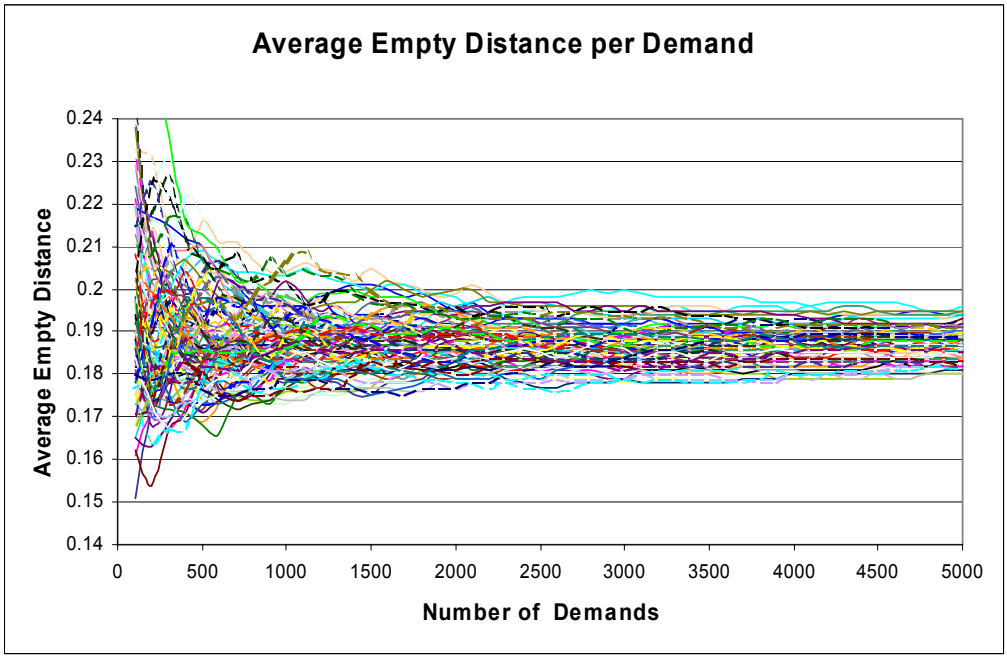


Figure 4.3 Average waiting time and empty distance vs. number of demands

4.4 ANALYSIS OF EXPERIMENTAL RESULTS

As explained previously, the basic motivation to control the number of candidates for reassignment is the premise that computation time increases exponentially with the number of candidate demands. To verify this assumption, the computation times to solve local problems of the reassignment stage (optimization-based solution method) of varying sizes are shown in table 4.1. For each case (number of candidates), 1000 problem instances are generated and solved. As seen in Table 4.1 and Figure 4.4, the computation time increases exponentially with the number of candidate demands. Reducing the number of demands causes not only a decrease in the number of variables, but also reduces the number of constraints in the mathematical formulation, such as flow conservation constraints, and time window constraints. Referring to the first MIP formulation in Section 3.3.1, the number of variables is $(K+N)^2 + N$, where K represents the fleet size and N represents the number of candidate demands involved with a local problem. The number of constraints is $N^2+4N+2K$ except for the binary variable assumption in constraint (3) of the MIP formulation.

Turning next to one iteration of the dynamic decision problem, with a sequence of 2000 load requests, the number of cases (reassignment optimization problem instances) that have more than 16 candidates for reassignment is 223 cases, out of a total of 2000 optimizations. However, these 223 optimizations consume 6582 seconds out of the total 7656 seconds of computation time for the entire sequence, in spite of the 500 sec maximum computation time limit. Note that, in order to avoid the extreme case, if an optimization procedure consumes

more than 500 sec, the best solution so far is taken and the procedure is forcibly stopped. This means that only 11% of the optimization instances, those with a relatively large number of variables, consume 86% of the computation time. Therefore, if the number of variables for each local problem can be controlled, the computation time can be controlled. Of course, the question that arises then is how much degradation in performance results from limiting the number of candidates. This is addressed next by moving the cut-off time described in the previous section.

Table 4.1 Computation time vs. number of candidates

# of Demands	Computation Time (sec)	Std. Dev.
3	0.042	0.136
4	0.054	0.162
5	0.068	0.193
6	0.079	0.219
7	0.099	0.255
8	0.149	0.340
9	0.216	0.448
10	0.303	0.750
11	0.493	1.251
12	0.873	2.013
13	1.663	6.833
14	3.018	10.025
15	5.523	16.647
16	11.791	32.406
17	23.453	57.495
18	37.462	72.576
19	72.522	103.349

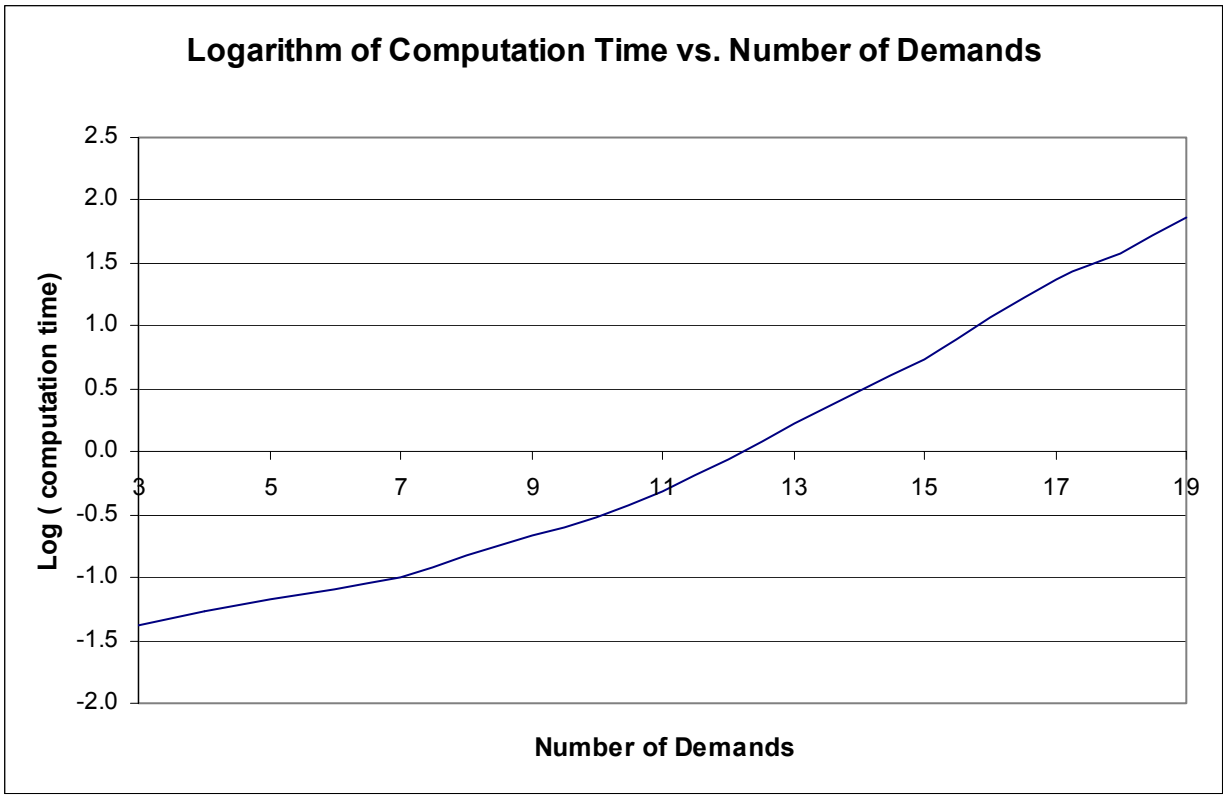


Figure 4.4 Logarithm of computation time vs. number of demands

4.4.1 Performance of the Dynamic Control of the Local Problem Size

Table 4.2 shows the performance of the dynamic control of the local problem size procedure. The number of ***max_cand*** for reassignment increases, the performance of the dispatching system improves in terms of empty distance and waiting time. If the number of candidates is not limited, the average number of demands in the system is 19.77, and an average of 11.33 demands are candidates for reassignment at each reassignment stage because the first demand in a vehicle's queue cannot be a candidate. Figure 4.5 illustrates the performance, which is represented in simulation time units, as the ***max_cand*** is increased from no reassignment (only implementing the first phase), to 3 and on to 10 candidates, with the full reassignment case also shown as a benchmark. In these results, when only maximum 3 demands are allowed for swapping and resequencing by optimal reassignment, it could reduce the empty distance 15% and waiting time 13%. However, when the candidate pool is relaxed from a maximum of 10 to allow full reassignment, only a small additional gain is observed in the performance measures, whereas the computation time to serve 2000 demands increases dramatically, from 446.65 seconds to 13381.50 seconds in spite of the 500 second maximum computation time limit. The exponential increase in total computation time as the ***max_cand*** increases is depicted in figure 4.6.

Table 4.2 Performance of moving time criterion cut procedure

max_cand	Empty Distance	Std. Dev.	Waiting time	Std. Dev.	# of Loads Accepted	# of Cand.	Comp. Time	Std. Dev.
no reassign	0.244	0.006	1.359	0.041	1927.8	0.000	13.90	1.48
3	0.207	0.006	1.183	0.054	1961.5	2.7144	104.40	3.14
4	0.199	0.005	1.146	0.045	1967.7	3.6645	119.95	3.27
5	0.192	0.006	1.114	0.054	1973.3	4.6111	138.85	4.16
6	0.188	0.005	1.095	0.047	1976.0	5.5474	167.10	5.52
7	0.185	0.005	1.080	0.048	1977.2	6.4478	202.15	7.76
8	0.180	0.004	1.065	0.041	1979.7	7.2991	255.65	13.82
9	0.177	0.006	1.055	0.046	1981.8	8.0632	337.00	18.60
10	0.175	0.005	1.051	0.041	1983.0	8.7674	446.65	32.01
Full	0.169	0.004	1.030	0.047	1985.8	11.330	13381.50	5949.69

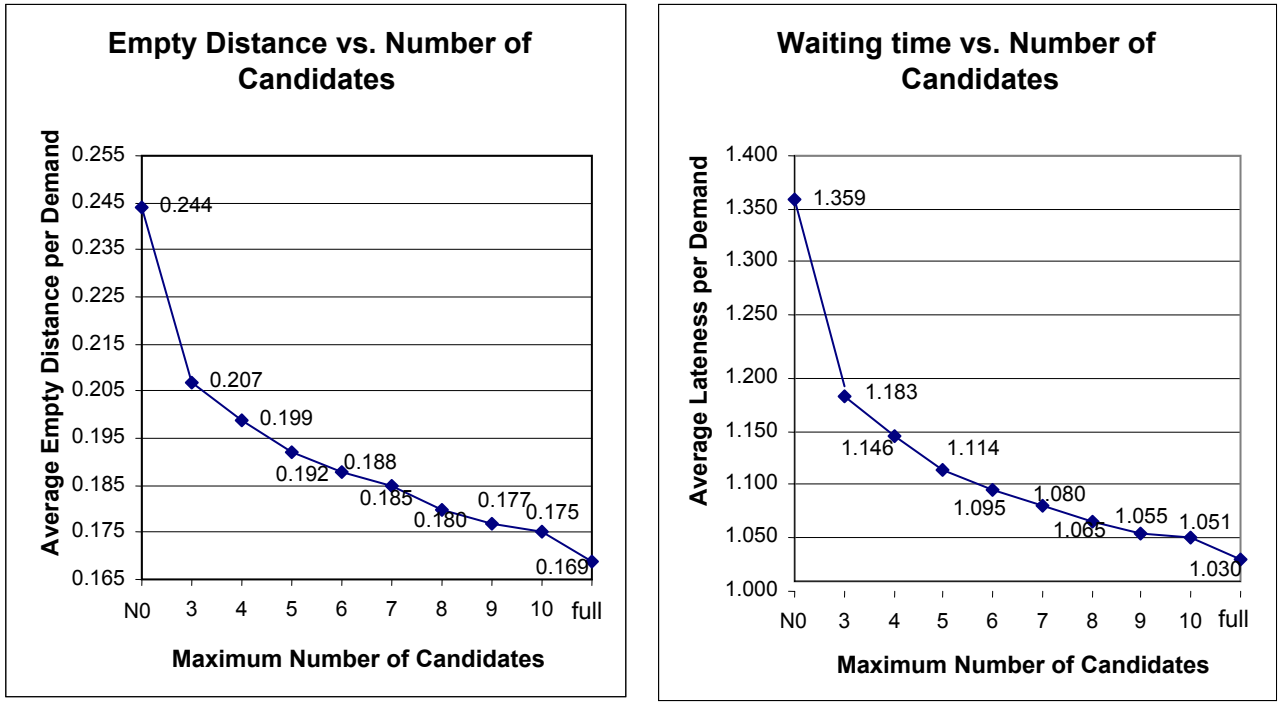


Figure 4.5 Performance of moving time criterion cut strategy

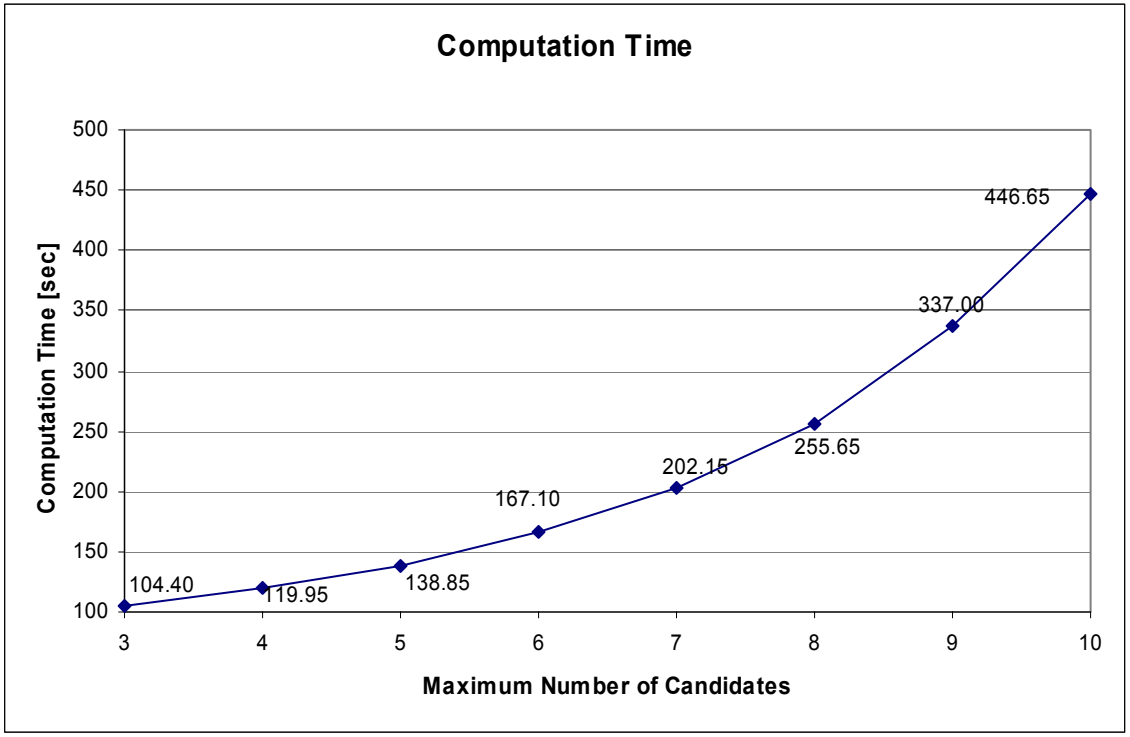


Figure 4.6 Computation time of moving time criterion cut strategy

Table 4.3 provides further insight into the role and effectiveness of the reassignment process, as the cut-off time is moved to reduce the number of candidates. For each maximum number of candidates, the number of reassignment problem instances is reported (note that the number is a little under 40,000, reflecting 20 iterations of 2000 loads each, some of which may have been rejected as non-feasible and some local problems do not have enough demands to be reassigned, hence the difference from 40,000). Also reported is the number of the optimization problem instances for which an improvement was attained over the initial solution obtained at the end of the first stage, and the corresponding percentage. Therefore, while with 3 candidate loads in the reassignment 26.43% of the instances result in some improvement, this percentage goes up to 52.86% when full reassignment is allowed; when the maximum is limited to 10 loads, about 49.43% of the cases result in improvement. It is notable that the first phase heuristic assignment results in a locally optimal solution in 47.16% of the cases. The average improvement in empty distance (conditional upon the improvement being positive) is 6.51% under full reassignment, dropping to 6.17% for the 10-candidate case, and 4.88% when only 3 loads are candidates for reassignment. Nonetheless, it is notable that the average improvement per local reassignment of 4.88%, which is only realized in 26.43% of the cases, translates into a global improvement (over the stream of demands) of about 15%. These results indicate that, even though the amount and frequency of improvement resulting from reassignment may be small with respect to the local problem, the global performance is improved significantly.

Table 4.3 Frequency of improvement

Maximum # of Candidates	# of reassignment problems	# of Cases with Improvement	Improvement Cases (%)	Average % of Improvement in Empty Distance per Reassignment (with Positive)
3	39230	10368	26.43	4.88
4	39354	12565	31.93	5.26
5	39468	14394	36.47	5.48
6	39522	16021	40.54	5.67
7	39545	17441	44.10	5.78
8	39595	18112	45.74	5.95
9	39637	18815	47.47	6.09
10	39662	19604	49.43	6.17
Full	38959	20594	52.86	6.51

4.4.2 Performance of the Merging Close Demands Strategy

In order to test the merging of closely spaced demands into single candidates for the reassignment optimization problem, two values of the threshold gap are specified: 0.05 and 0.03 (simulation units). This experiment is performed in conjunction with the previous strategy in order to evaluate the improvement for various sizes of the problem. As explained previously, consecutive demands in a vehicle's queue (after initial assignment), for which the empty distance is smaller than the threshold gap, are merged. Tables 4 and 5 summarize the results for gaps of 0.05 and 0.03, respectively. Note that the maximum number of candidates (first column) here refers to the merged loads treated as single units in the optimization program. For this reason, the actual number of elemental loads is greater on average (column 8).

Table 4.4 Performance results for gap threshold of 0.05

max_cand	Empty Dist.	Std. Dev.	Waiting time	Std. Dev.	# of Accept.	# of Cand	Actual # of Cand	Comp. Time	Std. Dev.
no reassign	0.244	0.006	1.359	0.041	1927.8	0.000	0.000	13.90	1.476
3	0.206	0.006	1.181	0.052	1961.8	2.714	2.749	98.70	3.028
4	0.199	0.005	1.144	0.045	1969.3	3.677	3.748	116.00	4.267
5	0.192	0.006	1.113	0.054	1972.4	4.616	4.729	138.15	3.360
6	0.186	0.006	1.090	0.053	1975.3	5.516	5.677	167.70	5.777
7	0.182	0.006	1.075	0.050	1977.9	6.374	6.585	204.85	6.098
8	0.179	0.005	1.067	0.047	1980.6	7.157	7.426	263.60	10.738
9	0.176	0.005	1.057	0.044	1982.2	7.833	8.159	345.50	23.180
10	0.175	0.005	1.054	0.049	1984.5	8.382	8.743	464.05	56.163
Full	0.170	0.004	1.029	0.034	1980.5	9.997	11.096	8078.10	3756.639

Table 4.5 Performance results for gap threshold of 0.03

max_cand	Empty Dist.	Std. Dev.	Waiting time	Std. Dev.	# of Accept.	# of Cand	Actual # of Cand	Comp. Time	Std. Dev.
no reassign	0.244	0.006	1.359	0.041	1927.8	0.000	0.000	13.90	1.476
3	0.207	0.006	1.191	0.050	1960.5	2.7125	2.730	98.60	3.218
4	0.198	0.006	1.146	0.046	1969.0	3.6626	3.692	115.25	3.508
5	0.192	0.005	1.113	0.039	1973.3	4.6133	4.662	138.55	3.677
6	0.186	0.005	1.086	0.039	1976.9	5.5381	5.610	168.80	5.116
7	0.183	0.005	1.077	0.050	1978.3	6.4291	6.520	204.05	6.533
8	0.179	0.005	1.062	0.048	1982.0	7.2378	7.356	259.70	11.779
9	0.177	0.004	1.051	0.033	1982.7	7.9676	8.102	344.40	19.586
10	0.175	0.005	1.051	0.049	1983.0	8.6178	8.776	455.35	38.516
Full	0.168	0.004	1.024	0.035	1986.1	10.704	11.135	9963.10	4186.461

Compared to the unmerged results of table 4.2, tables 4.4 and 4.5 reveal a slight improvement in performance for the same maximum number of candidates, and essentially similar computation time for the cases between 3 and 10 maximum candidates. The improvement in performance is due to the greater number of actual elemental candidates. Moreover, the results of the full optimization cases reveal that the merging strategy reduces the average number of candidates for reassignment from 11.330 to 9.997 and 10.704, respectively depending on the gap size. This reduction in the number of candidates reduces the computation time from 13381.5 seconds to 8078.1 and 9963.1, respectively, for about 40% and 26% improvement in computing time. As the threshold gap is increased, more demands are merged resulting in less computation time. However, this increase restricts the freedom of swapping of loads among the trucks, and should be traded-off accordingly.

4.5 SUMMARY

The MIP formulation of the local snapshot problem results in an *NP-hard* problem that cannot be solved in polynomial time for large problems. However, in order to use real-time information efficiently to make decisions about incoming requests – acceptance, assignment, scheduling –, these decisions should be made in a relatively short time. A hybrid dynamic decision approach, which consists of local heuristic rules for initial assignment and acceptance decision, coupled with formal optimization-based procedures for subsequent load reassignment decisions, is suggested and investigated in this chapter. In the first phase, the

acceptance decision and initial schedule of trucks are determined so as to respond to the customer. Then, some time is allowed to improve the schedule of trucks by swapping and re-sequencing the assigned loads using local optimization at the reassignment stage. In the second phase, however, computation time increases exponentially with the increase in problem size, hence motivating strategies to control this computation time. Two strategies are suggested for this purpose. Controlling the problem size using the moving time criterion cut and the merging of close demands strategy reduce meaningfully the computation time of a reassignment.

Extensive simulation experiments with these strategies highlight the tradeoff between computation effort and solution quality. The marginal increase of computing effort per unit increase of solution quality rises very rapidly when the local problem size is greater than a critical point, particularly in this problem setting, when a local optimum is required at every decision instance (full reassignment case). However, this critical point depends on the fleet size, demand arrival rate, and the time-window width of the requested demands. Therefore, a reasonable limit on the maximum number of candidates for reassignment can improve the initial schedule significantly within acceptable computation time. The merging of close demands further reduces the computing time in relatively large problems.

Chapter 5 Large Fleet Problems: Partitioning Strategies

5.1. INTRODUCTION

Chapter 5 introduces a problem setting managing a large fleet of vehicles in low to moderate demand arrival rate situations. The size of the fleet significantly affects the dispatching system in terms of the complexity of the local problem. The complexity of the local problem is a significant obstacle to the operation of fleet in the dynamic environment because the proposed policies are restricted to make a decision upon demand arrival before the next demand comes in. Furthermore, even with moderate demand arrival rate per vehicle, the large size fleet may cause the average inter-arrival time between consecutively requested demands to be too short to solve the local problems optimally before the next demand comes in. Therefore, the proposed dynamic operation policies are to solve the local problem as close to optimality as possible within the given time limitation that corresponds to the next demand arrival. Various procedures to managing a difficulty caused by the large size fleet are developed and tested such as partitioning strategies and local a problem control procedure. to reduce the local problem size.

Questions may arise on what to do if the solution of a given local problem has not been obtained yet (i.e., the solution procedure is still running) and the arrival of a new demand requires an immediate acceptance/rejection response, a decision that would typically depend on the result of the previous local problem being solved. Note that, under a given stochastic demand arrival process, one

cannot guarantee that local solution procedures will always be within this (random) time limit with probability one. The percentage of the cases (called 'violations'), for which the decision procedure computation triggered by a demand arrival cannot be completed prior to the next demand arrival, is reported in the simulation experiments conducted in this chapter. A policy to tackle this issue is explicitly discussed in Chapter 6 along with the response time concept.

The objective of the fleet management for the problem setting of this chapter is to find a policy minimizing the overall cost, $V \approx \min_{\pi \in \Pi} \sum_{i}^{A_i < T} \varphi_i^{\pi}$, and the main measurement is the average empty distance to serve a demand $\sum_{i}^{A_i < T} \varphi_i^{\pi} \Big/ \hat{N}$. In a low to moderate demand situation considering the ability of the fleet to serve the dynamically requested demand, assignment decisions may dominate the overall profit because all feasible demands can be served as addressed in Chapter 3.

The heuristic approach (Regan et al., 1996a) presented in Section 3.3.2.1 is employed as a benchmark policy, though only the initial assignment procedure is applied upon demand arrival. As an additional benchmark policy, the ***RAPID-SL*** algorithm (Powell, Snow, and Cheung (2000)) is applied to the same problem specification with various demand arrival rates. The details of this algorithm along with the simulation results are presented in Section 5.5.

5.2 DYNAMIC OPERATION POLICIES

The basic scheme to solve this large fleet problem follows the logic presented in Section 4.2.1. The initial assignment stage employs the first of the two methods, discussed in Section 4.2.1. A MIP formulation consisting of one vehicle and its demands, including the newly requested demand, is solved for each vehicle repeatedly over the entire fleet of vehicles, to identify the vehicle that could serve the new demand with least additional cost. The optimization-based reassignment procedure may require impractical heavy computing time to solve the large fleet problem. Thus, various partitioning strategies are introduced to select a manageable subset of high-potential vehicles with their associated demands for the reassignment procedure. This sub-problem is then solved optimally. Even though the resulting schedule is not the optimal solution for the entire fleet of vehicles, the reassignment procedure is intended to provide a good performance over the long run.

In a dynamic environment, even though a new instance of the local problem is obtained upon the arrival of a new demand, it is generally not necessary to regenerate the schedule of all the vehicles to obtain a near-optimal solution (assuming the existing schedule was near 'close-to-optimal solution for the given loads). This is particularly the case in large fleet problems because the updated local problem differs only in a minor way from the previous local problem instance: only a few trucks may have served their scheduled jobs, and only one new demand is added to the system during the inter-arrival interval separating the two problem instances.

5.3 PARTITIONING STRATEGIES

Several partitioning strategies are introduced in this section. These strategies are designed to select high-potential candidate vehicles for the reassignment procedure while keeping the computation time below a certain limit. The strategies are divided into two categories: 'fixed partitioning' and 'variable partitioning'. Within each category, several strategies are developed. The advantages and disadvantages of these categories and strategies are discussed. In addition, the approach presented in Section 4.2.1 to limit the number of candidate demands is applied to control the computation time.

5.3.1 Fixed Partitioning Strategies

In this category, the set of vehicles is partitioned a priori into mutually exclusive and collectively exhaustive groups that remain unchanged over the operational horizon. These subsets may be managed by dispatchers as though they are independent fleets. Two strategies for operating the fleet under a given fixed partition are introduced: 'round robin partitioning' and 'best group partitioning'. These strategies support relatively easy management of all trucks, as each dispatcher would keep close contact with the drivers in a group. A demand assigned to a group, however, cannot be reassigned to another group. This restriction limits opportunities to improve the schedule.

The 'round robin strategy' is the simplest operational strategy for use in conjunction with a fixed partitioning. The fleet is divided into fixed groups. A new demand is assigned to a vehicle in the first group, and the next demand to a

vehicle in the second group, and so on sequentially. The initial assignment and reassignment procedures are applied only to vehicles (and their respective assigned loads) within the group to which the demand has been assigned. Thus, it is expected that the computation time required in the initial assignment stage will be short compared to other policies because the dispatcher considers only one subset of vehicles at each decision instant.

Round robin partitioning

Step 0 *The vehicles are divided into* ***G*** *fixed groups, variable i =1*

Step 1 *When a new load comes in, assign the new load to group i*

Step 2 *Conduct initial assignment procedure within group i*

Step 3 *Conduct reassignment procedure within group i*

Step 4 *if the new load is the last demand during the time horizon then stop*

o.w. if i =G, i =0

Step 5 *i=i+1 go to step1*

The 'best group strategy' allows more flexibility in the selection of a group. Whenever a new demand comes in, the initial assignment procedure examines vehicles sequentially in the entire fleet. This procedure identifies a vehicle that requires the least additional empty movement to serve the new demand. Note that, the initial assignment procedure explores all vehicles in the fleet. For the remaining strategies including all the variable strategies employs this initial assignment procedure. Reassignment, however, is applied within the pre-determined group of vehicles to which the new demand is assigned. A load may not be assigned to a vehicle that belongs to another group.

Best group partitioning

Step 0 *The vehicles are divided into fixed groups*

Step 1 *when a new load comes in, conduct initial assignment over the full fleet of vehicles*

Step 2 *the candidate vehicles for the reassignment are the pre-determined group of vehicles for which the new demand is assigned*

Step 3 *if the new load is the last demand in the time horizon then stop*

Step 5 *go to step1*

5.3.2 Variable Partitioning Strategies

The other category of strategies consists of variable partitioning strategies. The main difference from the above category is that the candidate vehicles are selected at every decision epoch repeatedly, instead of using fixed groups. Thus, it is possible to reassign demands to vehicles drawn potentially from the entire fleet. Under variable partitioning strategies, the dispatcher needs to manage the entire fleet.

The first strategy considered is 'random partitioning.' After the initial assignment, the reassignment procedure is applied with randomly selected candidate vehicles and the vehicle to which the new demand has been assigned at the initial assignment stage ('initial vehicle'). In other words, a total of ***groupsize*** ($0 \leq groupsize \leq K$, predetermined parameter) vehicles are selected to define a local snapshot problem for the reassignment stage at every decision epoch. It is expected that the larger ***groupsize*** is, the better the solution obtained; however, it requires more computational effort.

Random partitioning

Stpe 0 groupsize: *predetermined parameter; the number of candidate vehicles*

Step 1 *whenever a new load comes in, apply initial assignment procedure over the entire fleet of vehicles*

Step 2 *Select the initial vehicle and* ***(groupsize-1)*** *vehicles randomly for the candidate of the reassignment*

Step 3 *Conduct reassignment with the selected vehicles and their associated assigned demands*

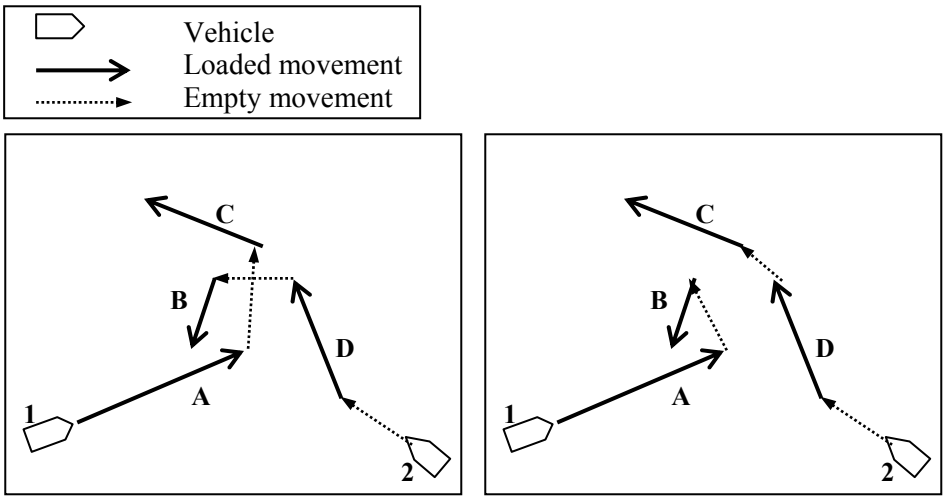
Step 4 *if the new load is the last demand in the time horizon then stop*

Step 5 *go to step1*

The second one is the 'geographic partitioning' strategy. The subset of vehicles selected to solve the MIP snapshot is determined on the basis of geographic proximity, as an indicator of likely potential as candidates for improved operation.

To illustrate the geographic proximity concept used here, consider two vehicles with the delivery schedule shown in Figure 5.1, demand **C** has been assigned to vehicle 1 as a result of initial assignment. In the figure, solid lines represent loaded movements and dotted lines show empty movements. Swapping the two demands **B** and **C** between vehicles reduces the total empty distance. This reduction is obtained by pairing two 'close' demands, **C** and **D** by connecting destination of one demand and origin of the other demand. Even though this possibility depends on the respective locations of the other demands and associated time-windows, selecting 'close' vehicles to the initial vehicle as

candidates for reassignment is expected to have high potential for reducing total empty distance.



(a) Before reassignment (b) After reassignment

Figure 5.1 An example of the close two vehicles

The precise 'proximity' between the initial vehicle and another vehicle is defined as follows. Distances between all possible parings among the demands associated with two vehicles are examined except for the pairings involving the first demand as the later demand of the pair (origin of the first demand in a vehicle queue to the destination of other). In addition, if a vehicle is idle, the distances between the current position of this vehicle and the origins of the demands in the other vehicle's queue are measured. Out of those distances, the minimum value will be taken as the proximity measure between the two vehicles. Because some pairings may be infeasible due to time-window restrictions, a simple time-window checking step is added such that the latest pickup time of the later demand of the pair should be greater than the earliest possible scheduled

pick-up time. The precise specific calculation of the 'proximity' between two vehicles are as follows.

Proximity (vehicle 1, vehicle2)

Step 0: *definition*

Vehicle #1: initial vehicle

Vehicle #2: target vehicle

t = current time

$\boldsymbol{o}(q_i^k)$ = *origin of the i* th *demand in vehicle k*

$\boldsymbol{d}(q_i^k)$ = *destination of the i* th *demand in vehicle k*

$\boldsymbol{o}^k$ = *current location of vehicle k*

*st(**d**(i)) = scheduled time when the vehicle reaches at the destination of load i*

$\tau^+(i)$ = *the latest pickup time of demand i*

$|q^k|$ = *queue length of vehicle k*

*D(**d**(i), **o**(j)) = distance between origin of load j and destination of load i*

Step 1: *Calculate* $F(q_x^1, q_y^2)$ for each pairing

$F(q_x^1, q_y^2)$: *If* $|q^2| \geq 1$, $F(q_x^1, q_y^2) = \{ D(\boldsymbol{o}(q_x^1), \boldsymbol{d}(q_y^2)) \}$
for all x = { 2, 3, ...,| q^1*|}, y = {1, 2, ..., |*q^2*|}*

If $|q^2| = 0$, $F(q_x^1, q_y^2) = \{ D(\boldsymbol{o}(q_x^1), \boldsymbol{o}^2) \}$
for all x={ 2, 3, ..., |q^1*|}*

Step 2*: Feasibility check:* $F(q_x^1, q_y^2)$ *takes Big value if the following conditions are not satisfied*

If $|q^2|=0$ $t + D(\boldsymbol{o}(q_x^1), \boldsymbol{o}^2) \leq \tau^+(q_x^1)$ *for all x = { 2, 3, ..., |*q^1*|}*

o.w. if y=1, st($d(q_y^2)) + D(\boldsymbol{o}(q_x^1), \boldsymbol{d}(q_y^2)) \leq \tau^+(q_x^1)$

$$\text{if } y \neq 1,\ t + D(\boldsymbol{o}(q_y^2), \boldsymbol{d}(q_y^2)) + D(\boldsymbol{o}(q_x^1), \boldsymbol{d}(q_y^2)) \leq \tau^+(q_x^1)$$

$$\text{for all } x = \{2, 3, \ldots, |q^1|\},\quad y = \{1, 2, \ldots, |q^2|\}$$

Step 3: *Calculate* $G(q_x^1, q_y^2)$ for each pairing

$$G(q_x^1, q_y^2) = \{ D(\boldsymbol{d}(q_x^1), \boldsymbol{o}(q_y^2)) \} \text{ for all } x = \{1,2, \ldots, |q^1|\},\quad y = \{2,3, \ldots, |q^2|\}$$

Step 4: *Feasibility check:* $G(q_x^1, q_y^2)$ *takes Big value if the following conditions are not satisfied*

If $y = 1$ $\quad st(\boldsymbol{d}(q_x^1)) + D(\boldsymbol{d}(q_x^1), \boldsymbol{o}(q_y^2)) \leq \tau^+(q_y^2)$

o.w. $\quad t + D(\boldsymbol{o}(q_x^1), \boldsymbol{d}(q_x^1)) + D(\boldsymbol{d}(q_x^1), \boldsymbol{o}(q_y^2)) \leq \tau^+(q_y^2)$

$$\text{for all } x = \{1,2, \ldots, |q^1|\},\quad y = \{2,3, \ldots, |q^2|\}$$

Step 5: *Proximity(vehicle 1, vehicle 2)=* $\min(F(q_x^1, q_y^2),\ G(q_x^1, q_y^2))$ *for all x,y*

The 'geographic partitioning' strategy selects the initial vehicle and remaining (***groupsize*** –1) vehicles in ascending order of this proximity measure.

Geographic partitioning

Step 0 ***groupsize:*** *the number of candidate vehicles*

Step 1 *whenever a new load comes in, initial assignment procedure is applied over the entire fleet of vehicles*

Step 2 *Calculate the proximity between the initial vehicle and vehicle k(k =1,…,K except for the initial vehicle)*

Step 3 *Select the initial vehicle and* ***(groupsize-1)*** *vehicles according to the proximity measure (ascending order) for the candidate vehicles*

Step 3 *Reassignment procedure is applied with the selected vehicles and their associated assigned demands*

Step 4 *if the new load is the last demand in the time horizon then stop*

Step 5 *go to step1*

To enhance the potential of the vehicles (and loads) included in the solution subset, and to compensate for the possible unbalanced vehicle selection over time (using solely geographic criterion), a 'hybrid partitioning' strategy is introduced to select a certain fraction of the candidates randomly. The 'proximity' of a vehicle (target vehicle) to the initial vehicle tends to improve as the number of demands in the target vehicle's queue increases, because the number of possible points for the 'proximity' calculation procedure increases with queue length. Therefore, the geographic partitioning strategy tends to select vehicles with long task queues, causing an unbalanced selection situation. In other words, vehicles with long queues tend to be selected repeatedly as reassignment candidates over time. In addition, the size of the local problems arising from geographic partitioning tend to increase compared to the 'random partitioning' strategy, potentially causing computational problems. The 'hybrid partitioning' strategy compensates for this problem by selecting some of the vehicles randomly, while the others are selected according to the 'geographic partitioning' strategy.

Hybrid partitioning

Step 0 *definition*

groupsize: *the number of candidate vehicles*

r: *the number of candidate vehicles to be randomly selected*

Step 1 *whenever a new load comes in, conduct initial assignment over the entire fleet of vehicles*

Step 2 *Calculate the proximity of vehicle k (to the initial vehicle) where k =1,…,K except for the initial vehicle*

Step 3 *Select* ***r*** *vehicles randomly including the initial vehicle, and the remaining vehicles are selected according to the proximity measure (ascending order)*

Step 3 *Reassignment procedure is applied with the selected candidate vehicles and their associated assigned demands*

Step 4 *if the new load is the last demand in the time horizon then stop*

Step 5 *go to step1*

In addition to the partitioning approach, the procedure presented in Section 4.2.1 to control the local problem size is applied and tested along with the best partitioning policy identified in the simulation experiments.

5.4 SIMULATION EXPERIMENTS

5.4.1 Simulation Framework

To evaluate the performance of the strategies developed in the preceding sections, a set of simulation experiments is designed. This chapter employs real scale units instead of the simulation unit used in Chapter 4. The geographical area for this simulation experiment is a 100 miles by 100 miles square region (which is same as 2 by 2 simulation units). The average loaded distance of a demand is then approximately 52 miles. It is assumed that the trucks have a constant speed of 50

mph. The mean inter-arrival time between consecutive demands per vehicle ($1/\lambda$) is specified as 90 minutes. With a fleet of 100 trucks, the dispatcher receives demand calls every 0.9 min (54 seconds) on average, and a truck would serve on average 10 demands per day.

The length of the time-windows of a demand ($\tau_i^+ - \tau_i^-$) is distributed uniformly between 2 and 4 hrs. This provides enough flexibility to construct routing schedule with multiple legs since the longest haul length is 141.4 mile, which can be served within 3 hours. In order to obtain a statistically significant result, fleet operations are simulated for 10 days. For this purpose, ten random seeds generate ten different demand streams. Each demand stream, which consists of about 1000 demands, represents the delivery requests in a day.

5.4.2 Numerical Results of the Partitioning Strategies

The evaluation measures are presented in the form average values to serve a load. For example, in Table 5.1, the 'initial assignment only (**IO**)' policy results in an average of 11.78 miles of empty distance to pickup a load, with a standard deviation of 0.42 miles. The customer waits for the service (waiting time: ($\delta_i - A_i$)) for 117.7 minutes with 2.70 minutes standard deviation. These standard deviations are calculated over 10 average values, while the standard deviations of the computation times are calculated on the N (≈ 1000) instances for each iteration, and averaged over 10 iterations.

$$\sigma_{empty\ (waitTime)} = \sqrt{\frac{\sum_{i=1}^{10} (\mu_i - \bar{\mu})^2}{9}}$$

$$where \quad \mu_i = \frac{\sum_{j=1}^{N} x_j}{N} \quad \textbf{for each iteration} \quad \textbf{i (i = 1, ..., 10)}$$

$$\sigma_{comTime} = \frac{\sum_{i=1}^{10} \sigma_i}{9}$$

$$where \quad \sigma_i = \sqrt{\frac{\sum_{j=1}^{N} (t_j - \bar{t})^2}{N-1}} \quad \textbf{for each iteration} \quad \textbf{i (i = 1,...10)}$$

The measure in which the standard deviations of the computation times are calculated reflect the concerns that the computation times for each decision epoch meet the problem's dynamic operational requirement. In other words, the decision procedure computation triggered by a demand arrival should be completed prior to the next demand arrival. The last column of the table, "violations", denotes the percentage of the cases where the requirement is violated.

The numerical results begin with Table 5.1, which presents the performance of one of the benchmark policies, the 'Initial assignment Only (**IO**)' policy. The violation percentage of the **IO** policy is 3.43%, indicating that 3.43% of the demands arrive during the execution of the decision procedure triggered by the previous demand, which takes 2.06 seconds on average raging from 1.16 seconds to 4.4 seconds.

The next category consists of the fixed partitioning strategies. The 'Round robin' (**RR**) strategy requires very short computation time in the initial assignment stage since the initial assignment procedure considers only a group of vehicles. However, even after reassignment, the **RR10** strategy, 'Round Robin' partitioning with 10 ***groupsize*** (a group consists of 10 vehicles) requires a much longer average empty distance to serve a load (20.2 miles) than the **IO** strategy (11.75 miles). In addition, the **RR10** strategy shows a rejection rate of 0.84%. Hence, considering the entire fleet of vehicles at the first stage provides considerable advantage to the carrier. The reassignment computation time in the sixth column increases dramatically as the ***groupsize*** increases from 10 to 20 in the 'Best group' strategies (**BR10**, **BR20**). The average computation time spent at a decision epoch including initial assignment and reassignment procedures under the **BR20** strategy is even longer than the average inter-arrival time of the demands, resulting in a violation rate of 25.3%. It is notable that the initial assignment computing effort is linearly proportional to the number of vehicles considered.

Table 5.3 presents the performance results of the variable partitioning strategies. In the table, the naming rules of the proposed strategies are as follows. **RP10** represents the 'Random partitioning' strategy with ***groupsize*** 10, in which 10 trucks are selected for the reassignment procedure. Furthermore, **H20-10** represents the 'Hybrid partitioning' strategy with 10 ***groupsize*** and the percentage of vehicles that are randomly selected is 20 %.

The initial assignment computation times in the fourth column, show relatively very small standard deviation compared to those of the reassignment computation times over all the proposed strategies. For example, **RP20** spends on average 3.11 seconds during the reassignment stage, which is not much longer than the initial assignment computing time (average 1.93 seconds). However, the standard deviation of the reassignment procedures over about 1000 applications triggered by demand arrivals and averaged over 10 iterations is 18.59 seconds. The latter indicates that the reassignment procedure consumes a very long time at some decision epochs. In contrast, the required computing efforts for the initial assignment procedures are stable, reflected in the 0.22 seconds standard deviation. That is, the dispatcher can guarantee a quick response to a customer with regard to an acceptance decision using the initial assignment.

Regarding the 'Random partitioning' (**RP**) strategies, the increase of the ***groupsize*** parameter (from 10, 15 to 20) improves performance in terms of the average empty distance. This increase, however, also requires much longer computing time to solve the reassignment MIP models involving the ***groupsize*** of candidate vehicles and results in a large number of violations. Thus, it is not desirable to increase the ***groupsize*** over 20. Among the random partitioning strategies, the **RP20** is chosen as the best strategy because it shows minimum empty distance with an acceptable computation time and violation rate.

The 'geographic partitioning' (**GP**) strategies outperform the **RP** strategies with respect to empty distance. However, for the same ***groupsize,*** the **GP** strategies tend to consider a larger number of candidate demands for a given

reassignment problem instance (column 6) than that of **RP** strategies. Thus, **GP** strategies require much longer computation time than the **RP** strategies, because **GP** strategies are inclined to select vehicles with long task queues, causing an unbalanced selection problem.

The 'Hybrid partitioning' strategies dominate the **GP** strategies in terms of empty distance and computation time. Furthermore, they outperform **RP** strategies with respect to the empty distance. The performance of a partitioning strategy in a local sense can be measured by the summation of the improvements in terms of empty distance during each reassignment procedure. The local improvements for the reassignment procedures indicate that the **GP** strategies perform best without considering the computation time. For example, the **RP15**, **H50-15** and **GP15** strategies show that the total empty distance reductions for the reassignment procedures are 2887.8 miles, 3988.8 miles, and 4542.0 miles, respectively. However, these values are based on the scheduled empty distance measured at each decision epoch. The actual traveled empty distance of the vehicles (correct measure of the performance) shows that the 'hybrid partitioning' strategies perform best in spite of the better local performance of the **GP** strategies. This result can be explained by the unbalanced selection tendency of the **GP** strategy.

Table 5.1 Performance results of IO policy

Strategy	Empty distance (mile) **a**		Waiting time (min) **b**		Initial Assignment Computation time (sec) **c**		Reassignment computation time (sec) **d**		Violations (%)
Initial assignment only (**IO**)	11.75	*0.42*	117.7	*2.70*	2.06	*0.25*	0.00	*0.00*	3.43

Table 5.2 Performance results of the fixed partitioning strategies

Strategy		***groupsize***	Empty distance (mile) **a**		Waiting time (min) **b**		Initial Assignment Computation time (sec) **c**		Reassignment computation time (sec) **d**		Violations (%)
Round Robin Partitioning	**RR10**	10	20.20	*0.58*	96.5	*3.01*	0.21	*0.04*	0.24	*0.74*	0.67
	RR20	20	14.55	*0.42*	91.8	*3.13*	0.38	*0.05*	3.66	*18.72*	4.62
Best group Partitioning	**BR10**	10	11.18	*0.42*	116.9	*3.61*	1.98	*0.23*	2.10	*16.12*	5.15
	BR20	20	10.62	*0.43*	115.3	*2.87*	1.97	*0.23*	66.56	*152.3*	25.30

a,b : Standard deviation over iterations
c,d : Average standard deviation over N demands

Table 5.3 Performance results of the variable partitioning strategies

Strategy		Empty distance (mile) a		Waiting time (min) b		Initial Assignment Computation time (sec) c		Reassignment computation time (sec) d		# of candidate demands	Violations (%)
Random Partitioning	RP10	9.75	*0.49*	108.5	*3.13*	1.95	*0.21*	0.17	*0.28*	11.07	3.63
	RP15	9.05	*0.42*	106.6	*4.12*	1.95	*0.22*	0.75	*3.37*	15.93	4.52
	RP20*	8.70	*0.44*	104.4	*2.51*	1.93	*0.22*	3.11	*18.59*	20.62	6.82
Geographic Partitioning	GP10	9.20	0.37	113.2	*4.44*	2.02	*0.26*	9.48	*41.03*	19.48	8.91
	GP15	8.91	0.42	112.8	*4.17*	2.00	*0.24*	64.47	*140.22*	28.65	28.42

Table 5.3 Continued

Strategy		Empty distance (mile)	a	Waiting time (min)	b	Initial Assignment Computation time (sec)	c	Reassignment computation time (sec)	d	# of candidate demands	Violations (%)
Hybrid Partitioning	H10-10	8.81	*0.42*	109.9	*2.54*	1.96	*0.24*	2.07	*12.87*	18.38	5.98
	H10-15	8.38	*0.42*	106.0	*2.96*	1.94	*0.23*	19.22	*66.29*	26.12	16.00
	H20-10	8.64	*0.41*	107.7	*2.41*	1.93	*0.22*	1.13	*6.56*	17.28	4.98
	H20-15	8.34	*0.37*	105.4	*2.84*	1.93	*0.24*	12.76	*78.67*	25.19	12.96
	H30-10	8.74	*0.43*	106.2	*2.95*	1.94	*0.22*	0.72	*2.39*	16.23	4.49
	H30-15	8.33	*0.37*	104.7	*2.93*	1.94	*0.25*	6.29	*29.55*	23.39	9.42
	H40-10	8.85	*0.43*	106.6	*2.66*	1.95	*0.24*	0.59	*2.52*	15.45	4.27
	H40-15	8.39	*0.38*	103.9	*2.72*	1.94	*0.23*	4.74	*24.80*	22.42	8.55
	H50-10	8.81	*0.39*	106.5	*3.09*	1.94	*0.21*	0.46	*2.28*	14.51	3.96
	H50-15*	8.46	*0.35*	104.2	*3.64*	1.91	*0.21*	2.39	*11.01*	20.76	6.63
	H70-10	9.05	*0.36*	106.6	*2.45*	1.94	*0.22*	0.25	*0.39*	12.72	3.74
	H70-15*	8.57	*0.44*	104.1	*3.48*	1.93	*0.21*	1.11	*5.45*	18.21	5.09

a,b : Standard deviation over iteration
c,d : Ave. standard deviation over N demands

Figure 5.2 depicts the frequency that each vehicle is selected as a candidate for the reassignment procedure during a day. If all vehicles were selected evenly, each truck would be selected 150 times, because every MIP model during the reassignment stage involves 15 candidate vehicles out of 100 vehicles about 1000 times. As can be seen, vehicles are selected almost evenly when the **RP** strategy is applied (the last graph). The first graph indicates that the selection procedure of the **GP** strategy is biased. This biased selection procedure misses the opportunity of reassigning demands to diverse vehicles. The 'Hybrid partitioning' strategy compensates for the unbalanced selection problem, and exhibits the best performance.

Among various parameter settings, **H50-15**, which requires 8.46 miles of average empty distance to serve a load and average of 4.2 seconds (1.91 for initial assignment + 2.39 for reassignment) of computing time for an assignment decision procedure, and **H70-15** (8.57 empty moving miles, 3.04 computation time) dominate the **RP20** strategy with respect to both the computation time and the empty distance.

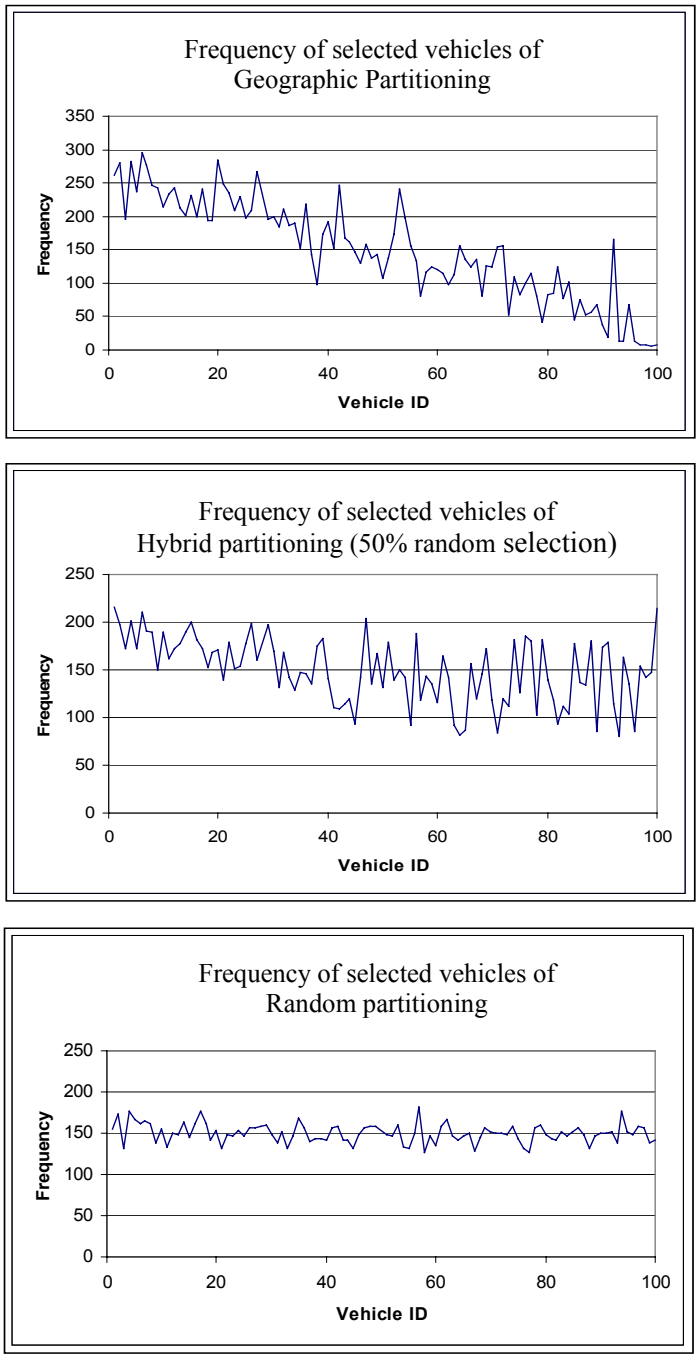


Figure 5.2 Selection Frequency of each vehicle for the reassignment procedure

The results of the dynamic problem size control procedure applied along with the 'Hybrid partitioning' strategies are presented in Table 5.4. Since this procedure is expected to reduce the computation time, it is applied to the strategies that show better performance with somewhat longer computation time. Thus, the ***groupsize*** is specified as 15 vehicles. Furthermore, the ***max_cand*** parameters are specified as the mean of the number of candidates, and the mean plus one standard deviation observed in the 'Hybrid partitioning' strategies.

Table 5.4 shows that controlling local problem size by the time cut criterion reduces the computation effort significantly for the reassignment stage. For example, **D40-22,** in which 40% of candidate vehicles are selected randomly and the ***max_cand*** is specified as 22, spends an average 1.54 seconds during the reassignment procedure. This result is competitive with the 4.74 seconds reassignment time of the **H40-15** strategy, which shows similar average empty distances (8.43 miles and 8.39 miles). Furthermore, **D40-22** dominates **H50-15**. In addition, **D50-20** dominates **H70-15** in terms of empty distance and computation time.

Compared with the benchmark (**IO**) strategy, in which only the initial assignment is applied, the **D40-22** strategy shows significant reduction of average empty distance from 11.75 miles to 8.43miles while the violation rate increase only 2.45% (from 3.43% to 5.88%). However, even with the best strategy in this chapter, violations are inevitable because some demands arrive almost simultaneously.

Table 5.4 Performance results of 'dynamic problem size control' applied to the 'hybrid partitioning'strategy

Strategy	Empty distance (mile)	**a**	Waiting time (min)	**b**	Initial Assignment Computation time (sec)	**c**	Reassignment computation time (sec)	**d**	# of candidate demands	Violations (%)
D30-23	8.44	*0.38*	104.37	*2.70*	2.02	*0.24*	2.26	*11.03*	21.27	6.61
D30-28	8.40	*0.45*	105.13	*3.16*	1.95	*0.23*	4.57	*22.05*	23.13	8.69
D40-22*	8.43	*0.40*	104.95	*2.33*	1.93	*0.21*	1.54	*7.02*	20.39	5.88
D40-25	8.41	*0.36*	104.88	*3.86*	1.94	*0.21*	2.64	*12.60*	21.72	7.00
D50-20*	8.56	*0.38*	104.55	*3.21*	1.95	*0.22*	0.91	*2.10*	18.48	4.99
D50-25	8.45	*0.42*	103.19	*3.12*	1.95	*0.21*	1.71	*7.28*	20.33	5.91

a,b : Standard deviation over iteration
c,d : Average standard deviation over N demands

The number of cases that reassignment procedure reduces the scheduled total empty distance of the fleet is an average of 503.9 times out of 1000 applications of the **D40-22** strategy (the average is taken over the 10 replications performed), and each reassignment procedure reduces on average 2.76% of empty distance compared to the initial schedule obtained in the first phase (the average is taken over 1000 applications). Global performance, however, shows 28.26% reduction compared to the **IO** strategy. This indicates that the initial schedules obtained during the initial assignment in **D40-22** are competitive (close-to-optimal) schedules. When a new demand arrives, the given schedule determined at the previous decision epoch is close-to-optimal so that a simple heuristic (initial assignment), which changes only one vehicle's schedule, can achieve a high quality solution. Therefore, in this dynamic environment, it is preferable to use the schedule of the previous decision epoch as an initial solution rather than to totally re-optimize the schedule. Note that this approach is particularly useful for large fleet size problems, where total re-optimization is near impossible due to the computational complexity.

5.5 COMPARISON WITH RAPID-SL

For the same class of problems, Powell, Snow, and Cheung (2000) proposed two optimization-based heuristic algorithms, ***RAPID-SL*** (Resource Allocation procedure for the Integrated Dynamic Assignment Problem-Single Label) and ***RAPID-ML*** (Multi-Label). Since the authors recommend the ***RAPID-SL*** algorithm in low to moderate demand arrival situations, the latter is

implemented for the problem of interest here in order to evaluate relative performance. A brief description of the ***RAPID-SL*** algorithm is first presented; additional detail is available in the cited reference.

A (single) label representing the attributes of a driver (or vehicle) following the completion of a task is iteratively updated in the algorithm. A new mathematical formulation is developed, which is relatively simple and able to handle a high degree of operational detail of the fleet management problem. However, the formulation contains a large number of subtour constraints and highly nonlinear constraints. The presence of those constraints is the main hurdle to solve the formulation. The ***RAPID-SL*** algorithm eliminates the nonlinear constraints by iterative updates of the label, and relaxes the subtour constraints. As a result, the problem becomes a pure network problem. This network problem was solved repeatedly to eliminate subtours by penalizing the arcs in the subtours using dual variables. This procedure iteratively updates the driver attributes by adjusting label and cost until no assignments have changed from the last iteration or pre-specified number of iterations. Nonetheless, there is no guarantee that all subtours will be eliminated in the ***RAPID-SL*** algorithm.

This algorithm is coded and implemented for the same dynamic problem with various demand request scenarios. Since ***RAPID-SL*** algorithm provides the solution approach for the local snapshot problem rather than the dynamic problem itself, two dynamic operational implementation strategies are developed in this thesis. The first strategy (strategy I) rejects the demands in the subtours regardless of the acceptance decision at the demand arrival instant, while the other strategy

(strategy II) holds the demands in the subtours, which are then include in the next local snapshot problem (triggered by the next demand arrival) as long as their time-windows remain feasible. However, the latter strategy still cannot guarantee that all accepted demands are served.

Table 5.5 Performance Summary of RAPID-SL algorithm in comparison with D40-22 strategy

	Inter arrival time/veh = 90min			Inter arrival time/ veh = 60 min		
	D40-22	RAPID-SL		D40-22	RAPID-SL	
		I	II		I	II
Total Empty Time	159.3	139.8	151.0	184.6	134.8	144.6
Total Loaded Time	1033.5	957.1	986.9	960.7	907.4	935.5
Total Idle Time	488.5	551.2	506.4	186.0	194.6	160.1
Total Empty Distance	7965	6990	7548	9232	6740	7231
Total Loaded Distance	51676	47855	49346	48034	45372	46776
# of served demands	1000	895	952	935	836	897
Profit	$28,016	$26,164	$26,786	$24,999	$24,743	$25,347

Table 5.5 presents the performance results of the ***RAPID-SL*** algorithm in comparison with the **D40-22** strategy presented in the previous section. The computation time of these dynamic operational strategies is not examined because the author of this thesis coded the algorithm. Nonetheless, the quality of the solution is comparable. The number of served demands indicates that the ***RAPID-SL*** algorithm rejects a fraction of demands under given demand arrival rates. Thus, a monetary measure, in which $1.2/mile for loaded distance (revenue) and

unit cost, $.57/mile for traveled distance (empty +loaded) are defined for fair and precise comparison. As seen in the table, **D40-22** is competitive with the ***RAPID-SL*** particularly in the less congested demand arrival situation. Note that, since the ***RAPID-SL*** implementation does not enforce that all the accepted demands are included in the current schedule, some demands accepted at previous decision epochs may be rejected when the scheduled is updated.

5.6 SUMMARY

In this chapter, hybrid dynamic decision policies along with various partitioning strategies are developed to solve large fleet dynamic routing and scheduling problems. The findings are as follows: The performance of the dispatching system is significantly improved as the number of trucks included in the initial assignment increases. Moreover, the initial assignment computation time increases almost linearly with the fleet size. Thus, it is recommended to included the entire fleet of vehicles in the initial assignment stage. This initial assignment changes only one vehicle's schedule compared with the schedule of the previous decision epoch, which is a close-to-optimal schedule. Thus, when selecting candidate vehicles during the reassignment stage, a proximity measure was developed in order to find high potential vehicles where the selected vehicles are 'close' to the vehicle to which the new demand has been assigned during the initial assignment. This selection process performs well particularly in a local sense. However, the 'Geographic partitioning' strategy using this measure failed to reassign demands to diverse vehicles in a global sense. Thus, this partitioning

strategy results in biased candidate vehicle selection and tends to increase the size of the local problems. To circumvent this problem, a 'hybrid partitioning' strategy combining 'geographic partitioning' with 'random partitioning' is developed. This strategy performs well with respect to computation time and overall solution quality. During the reassignment procedure, the 'dynamic problem size control' strategy described in Chapter 4 is applied, helping to reduce the average computation time and its variance, which maintaining solution quality.

In this chapter, the following dynamic operational restriction on solving the local problem has been imposed: computation at each decision epoch should be completed before the next demand arrival. However, it is difficult to predict and control the required computation effort for the multi-vehicle off-line MIP problem. Moreover, if demand requests arrive at the dispatcher in very rapid succession, it is impossible to always satisfy this dynamic operational restriction. Therefore, some violation cases (when the next demand arrives during execution of the decision process triggered by the current demand) were reported in the simulation experiments. Furthermore, as the arrival rate increases or a much larger size fleet is considered, violation cases would be more frequently observed. This has motivated the development of a dynamic decision policy that does not impose this restriction while satisfying various dynamic operational requirements, as discussed in the next chapter.

Chapter 6 Dynamic Adaptive Dispatching Strategies in Over-saturated Demand Situations

6.1 INTRODUCTION

This chapter focuses on the dynamic fleet management problems in over-saturated demand situations, i.e. when the demand for service exceeds the system's average capacity to provide delivery service within specified time-windows of the requested demands. In these problem settings, an appropriate measure to evaluate performance of the proposed policies is overall profit rather than average cost to serve a load because it is impossible to accept and serve all the demand requests. Thus, the overall objective is to find the optimal policy π, if such a policy exists, that maximizes total profit.

$$V^{\pi} = \sum_{i}^{A_i<T} D_i^{\pi} (R_i - \beta(\varphi_i^{\pi} + l_i))$$

Where V^{π} represents the total profit obtained when policy π is applied, in which the revenue is earned through the total loaded distance, R_i denotes the reward from the accepted load i, which is proportional to the haul-length of load i (l_i) such that $R_i = r * l_i$., where r represents the unit revenue per loaded mile. The cost is incurred on the total moving distance including both the empty and loaded movements. The unit transportation cost per traveled mile is denoted by β. Hence, the profit increases as the fleet serves more longer haul-length demands with less empty distance.

Another measure to evaluate policy performance is response time to a customer. This time is measured from the demand request time (A_i) to the time that a shipper receives an acceptance/rejection decision from the dispatcher.

In this problem, the relationship between solution algorithm execution time for a local snapshot problem and solution quality, measured by response time, is explicitly discussed. A dynamic operation policy adaptively implementing various assignment procedures is described, with the aim of fully utilizing computational resources without wasting any time that may be available between successive demand requests. In addition, application of intelligent acceptance decision and demand filtering procedures that consider system status and demand characteristics result in significant operational benefits.

6.2 ASSIGNMENT TECHNIQUES

The three load-truck assignment techniques introduced in Chapter 5, are applied to this problem. All the assignment techniques use the Mixed Integer Programming (MIP) model presented in chapter 3. The first assignment technique (*Assign I*) is the initial assignment procedure. The second assignment procedure (*Assign II*) reassigns existing loads in the system using the 'Hybrid partitioning' strategy, developed in Chapter 5. This strategy judiciously selects a subset of the vehicles in the fleet, with their associated assigned demands. The last assignment technique (*Assign III*) reassigns existing demands in the system using the 'Random partitioning' strategy that randomly selects the candidate vehicles for reassignment. The manner in which theses three assignment techniques are

applied in an online operational setting, and the events that trigger each respective application, are described in the next section.

6.3. DYNAMIC DISPATCHING STRATEGY

In typical adaptation of static algorithms to a dynamic operational setting, the arrival of a new demand triggers the solution procedure of the static snapshot problem generated from information on the current state of the system, such as the respective locations and status of the vehicles, as well as the respective locations and associated time-windows of the known demands. The solution resulting from this procedure provides a schedule for the vehicles in the fleet. The vehicles follow this schedule until the next schedule is generated from the solution of the next snapshot problem triggered by the arrival of a new demand.

Since the next demand arrival time is not known in advance, researchers have focused on minimizing the computation time (for the snapshot problem) in order to meet the dynamic operational condition that a decision process be completed before the next demand arrival. However, imposing an arbitrary pre-determined short computation time may waste computation capability between demand arrivals. The main motivation for limiting solution time is that a dispatcher should almost immediately inform the customer of the acceptance decision for a newly requested demand. However, in order to obtain a quality solution for this combinatorial problem, it is practically impossible to reduce the computation time to a second, and it is not necessary to do so. In this chapter, a maximum tolerable response time is specified to provide a customer with an

acceptance decision. This is the amount of time a customer is willing to wait before receiving an acceptance or rejection decision.

In order to satisfy this restriction and maximize computation capability within a given time, a Dynamic Adaptive Dispatching (*DAD*) strategy is developed. This strategy utilizes computation capability fully without wasting available time. Load arrival triggers the initial assignment procedure (*Assign I*). If no new demands arrive during the execution time of *Assign I*, and the new load is accepted, *Assign II* (which includes this new demand) is applied repeatedly until the next demand arrives. If the new load is rejected, *Assign III* is applied repeatedly to the existing loads in the system until the next demand arrives. The purpose of repeated application of *Assign III* is to allow consideration of new randomly selected vehicles in each application, in an effort to uncover additional improvement opportunities through local assignment and swapping across vehicles.

The maximum computation times of *Assign II* and *III* are pre-specified and an appropriate snapshot problem size (the number of candidate vehicles and demands) is defined according to this threshold, along the lines described in chapters 4 and 5.

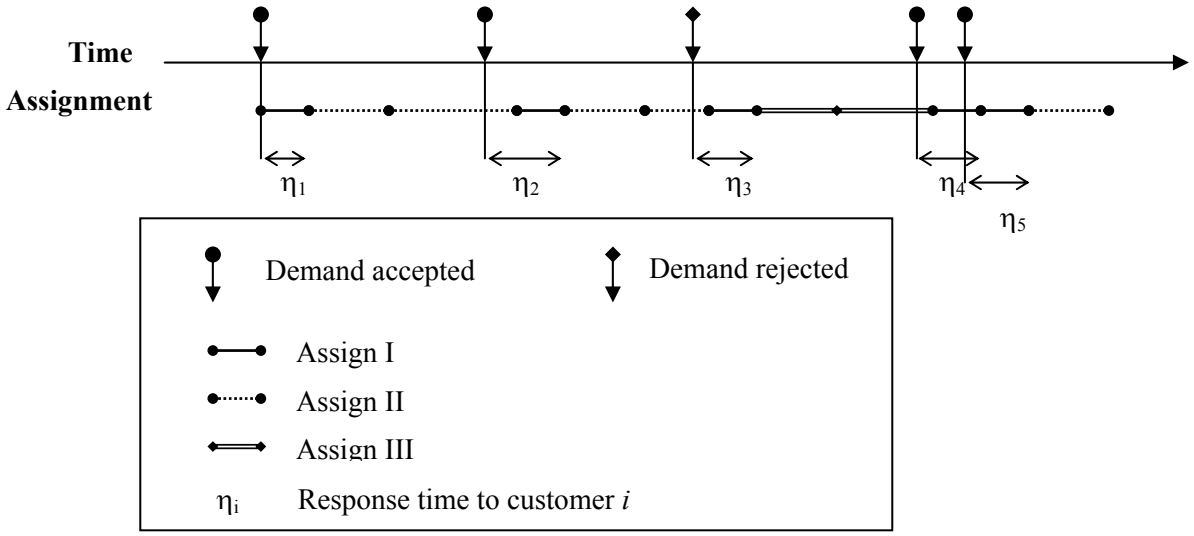


Figure 6.1 Implementation of DAD strategy for a dynamic problem

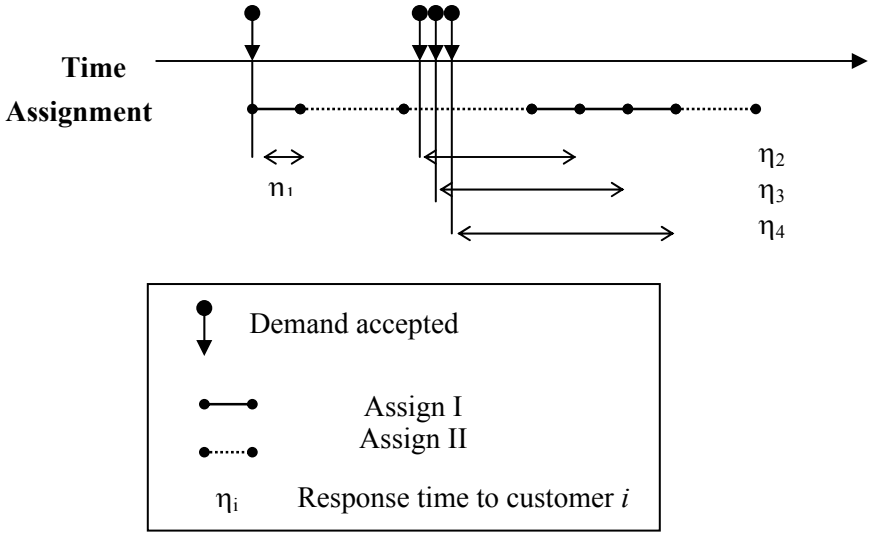


Figure 6.2 Example of worst-case response time

Figure 6.1 illustrates the implementation of the *DAD* strategy. As can be seen, the response time to a customer is generally the sum of *Assign I* execution time and part of the *Assign II* (or *Assign III*) execution time, depending to the next demand arrival instant. The worst case arises when several demands are requested almost simultaneously just after *Assign II* (or *Assign III*) starts. Figure 6.2 illustrates an example of such worst case, in which the fourth customer needs to wait for a period of η_4 in order to receive an acceptance decision.

Another computation time issue is the relationship between vehicle movement and assignment execution time. When a local snapshot problem is generated on the basis of the current state of the vehicles and the known but not served demands, it is important to avoid the conflict discussed in Section 3.3.2.2. Thus, for the assignment procedures, the maximum computation time for a single

application should be predefined, and taken into consideration when specifying the associated local problem. Thus, the assignment process uses updated vehicle locations and status information instead of current information, as discussed in Section 3.3.2.2.

The value of the allowed computation time of an assignment procedure is approximately a function of problem size. Therefore, the maximum allowable computation time parameter is defined as follows. In case of *Assign I*, the computation time parameter is specified as the maximum possible computation time based on data obtained from simulation experiments. The computation time of *Assign I* depends on the number of demands in the system. Hence, the parameter is adaptively modified depending on this value. Generally, the computation time of *Assign I* is relatively short, whereas *Assign II* and *III* procedures require relatively long computation times. Therefore, the computation time parameter is predefined, taking into consideration the tolerable response time to a customer. According to this value, the problem size of a local problem is defined. However, controlling problem size does not determine the computation time exactly. Thus, if the *Assign II* or *III* processes cannot obtain an optimal solution for the local problem within the maximum allowable time, the best solution found within the given time will be used.

6.4 FILTERING

Due to the time-windows associated with the demands, the number of demands in the queue of the system is limited. The 'holding capacity' is defined

as the maximum number of demands waiting for service in the system. High arrival rates (corresponding to highly congested conditions) cause the system to operate continually at this 'holding capacity'. When the number of accepted demands approach the 'holding capacity', it becomes difficult to find reassignment opportunities to reduce the empty distance because there is not enough room for swapping and re-sequencing existing demands, and most demands end up being served near the upper bound of their respective time-windows.

The filtering process aims to control the total number of demands in the dispatching system. Whenever a new demand arrives, the number of demands in the system (the accepted demands that have not been picked up yet) is considered. When the number of demands in the system is at or above a predefined threshold, all arriving demands are rejected out right. Only if the number of demands is below the threshold is the *Assign I* procedure applied. Otherwise, the demand is rejected outright. This filtering process will not be applied towards the end of the operation horizon because the dispatcher wants to serve as many demands as possible within a working horizon. Hence, after a certain time near the end of the horizon (to be determined on the basis of the service time and characteristics of the particular operation), all the feasible demands are accepted. Note that the 'holding capacity' does not represent the service capacity. The filtering process only limits the number of demands in the queue, not the number of served demands in a day. In fact, in some cases, the filtering process may allow a greater number of loads to be served over the horizon of interest.

In addition, an intelligent filtering decision process (*AddCost*) is developed. The *Assign I* initial assignment procedure provides a good measure for characterizing the potential of a new demand, namely the additional cost resulting from assigning the demand to a vehicle. The new demand will be accepted only if the additional cost is less than a certain level and the total number of demands in the system is less than the filtering threshold.

6.5 EXPERIMENTAL DESIGN

A simulation framework similar to the one presented in chapter 5 is employed to evaluate the performance of the algorithms. In this chapter, it is assumed that a company receives demands for 10 hours a day (T = 10 hours) and operates until the fleet completes delivery service for all accepted demands. Thus, the company works for about 15 hours a day to complete its delivery service for the jobs in the system. A single run (simulation of the performance of the fleet for one a day) takes almost the same time as real world time, namely 10 hours, as explained in Section 3.4.1. Thus, five iterations of the simulation experiments are conducted, simulating one week of operation.

The simulation experiments in this chapter assume a demand rate that produces an over saturated system. Therefore, the demand arrival rate is specified so that about half of the demands are rejected even with an efficient dispatching algorithm. This requires in an average inter-arrival time of successive demands of around 18 seconds. The maximum allowable computation time for *Assign II* and *Assign III* is specified as 10 seconds.

6.6 NUMERICAL RESULTS

First, in order to evaluate the *DAD* algorithm implementation process, the dynamic decision policy developed in Chapter 5 (named TPD two phase dispatching policy) is applied to the same demand stream. In this process, every new demand arrival triggers *Assign I* and *Assign II* sequentially. If a new demand arrives during the execution of an assignment procedure, *Assign I* for the new demand is applied after completing the current procedure (i.e. either when optimality is reached or a preset maximum execution time is reached, whichever comes first). Table 6.1 summarizes the comparison of the two alternative process implementations.

As shown in table 6.1, *DAD* generates more profit as a result of longer total loaded distance and a greater number of accepted demands served with less total empty distance. Thus, *DAD* improves average profit by 3.91% compared to the *TPD* benchmark. With regards to the response time, *DAD* exhibits a slightly longer average response time. However, the worst case (maximum value row) shows that *TPD* may require a much longer response time for some customers.

Table 6.1 Performance comparison of TPD and DAD

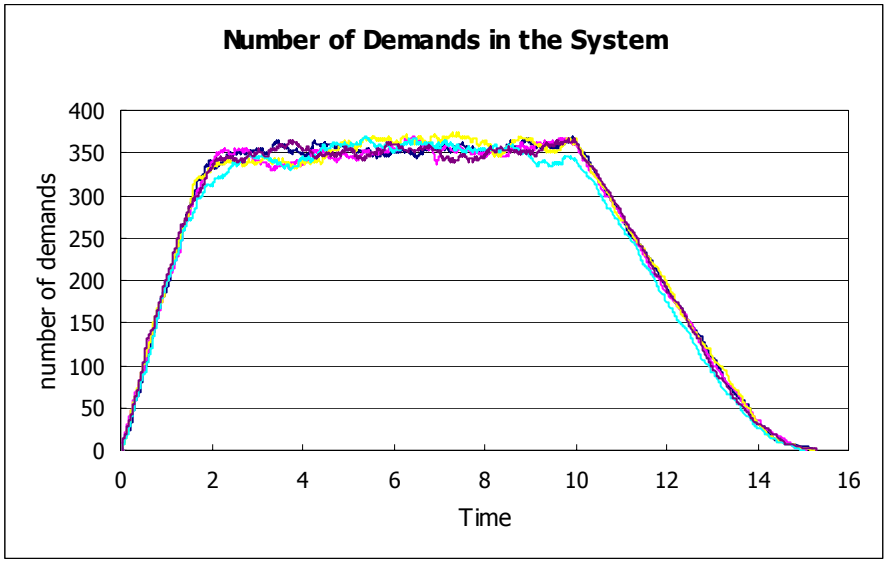
(a) Performance

	Iteration	Total Empty Distance [mile]	Total Loaded Distance [mile]	# of served demands	Waiting time [min]	Profit
TPD	1	11236.1	53724.8	1090	161.4	$18,442.1
	2	11520.0	53699.0	1062	161.0	$18,264.0
	3	11430.3	54034.7	1092	162.4	$18,526.6
	4	11628.6	52700.6	1047	160.8	$17,573.1
	5	11099.6	53609.6	1074	161.0	$18,447.3
average		**11382.9**	**53553.8**	**1073.0**	**161.3**	**$18,250.6**
DAD	1	10734.3	54346.8	1083	161.4	$19,119.9
	2	10896.5	54141.9	1067	161.5	$18,898.4
	3	10793.4	54597.3	1093	160.4	$19,244.1
	4	10849.0	53525.2	1055	160.0	$18,536.9
	5	10686.6	54146.5	1081	160.3	$19,021.0
average		**10791.9**	**54151.5**	**1075.8**	**160.7**	**$18,964.1**

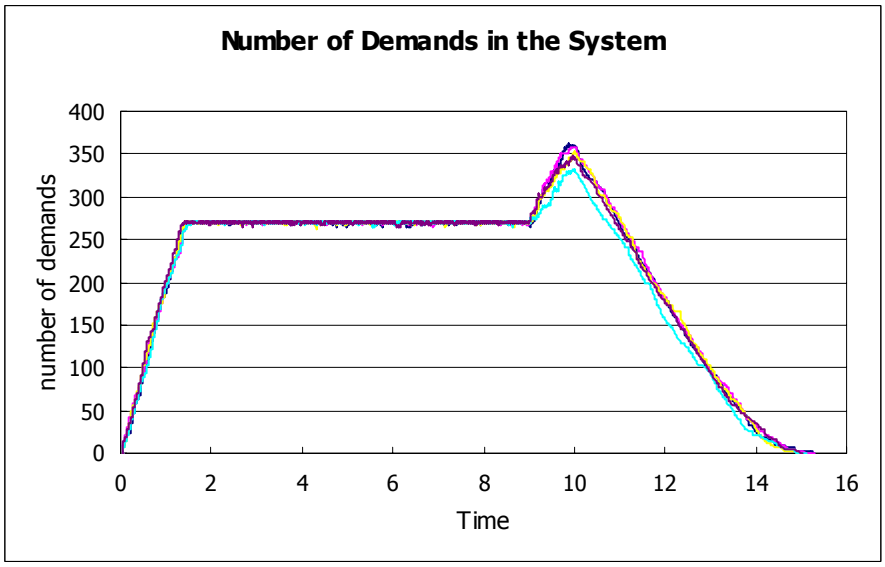
(b) Response time [sec]

	TPD	DAD
Mean	5.44	6.62
Standard Deviation	4.15	2.99
Minimum	1.25	1.25
Maximum	39.51	22.65

The evolution of the total number of demands in the system over time within a day is shown in Figure 6.3, first without application of the filtering process in Figure 6.3(a), then with filtering in Figure 6.3(b). The different lines correspond to five iterations (5 days with different demand stream realizations). The 'holding capacity' of this system is around 360 demands and the system reaches this level by the second hour of operation (Figure 6.3 (a)). After the 10^{th} hour, the company does not accept any new demands, and focuses on serving already accepted demands. Figure 6.3 (b) shows the effect of the filtering process with a threshold of 270 loads, so that all demands received when the system is at this level are rejected outright. If the number of demands in the system is below the 270 threshold, acceptance/rejection is determined in the basis of the *Assign I* procedure. This filtering process was applied until the 9^{th} hour. Limiting the number of loads in the system at any given time provides room for swapping and re-sequencing to improve a schedule. Examination of when loads are actually served without filtering reveals that a majority of the loads are served near the end of their respective pick up windows, as reported in Table 6.1 (Waiting time). When these constraints are binding for most loads, little slack exists to improve operation through swapping and rescheduling. The filtering process retains some flexibility without allowing vehicles to go idle, resulting in higher profit, as illustrated in the next set of results.



(a) Before Filtering



(b) After Filtering

Figure 6.3 Effect of filtering process

To identify the "optimal" filtering threshold, various filtering thresholds are implemented and the results are shown in Figure 6.4. In the figure, a dot represents the total profit for the particular iteration (day). Five iterations are presented for each algorithm, respectively. The line connects the average values obtained for each threshold. As can be seen, F270 shows the best performance. This level, which is 25% less than the 'holding capacity' provides room for reassignment, and appears to be the most effective filtering threshold considered in this search.

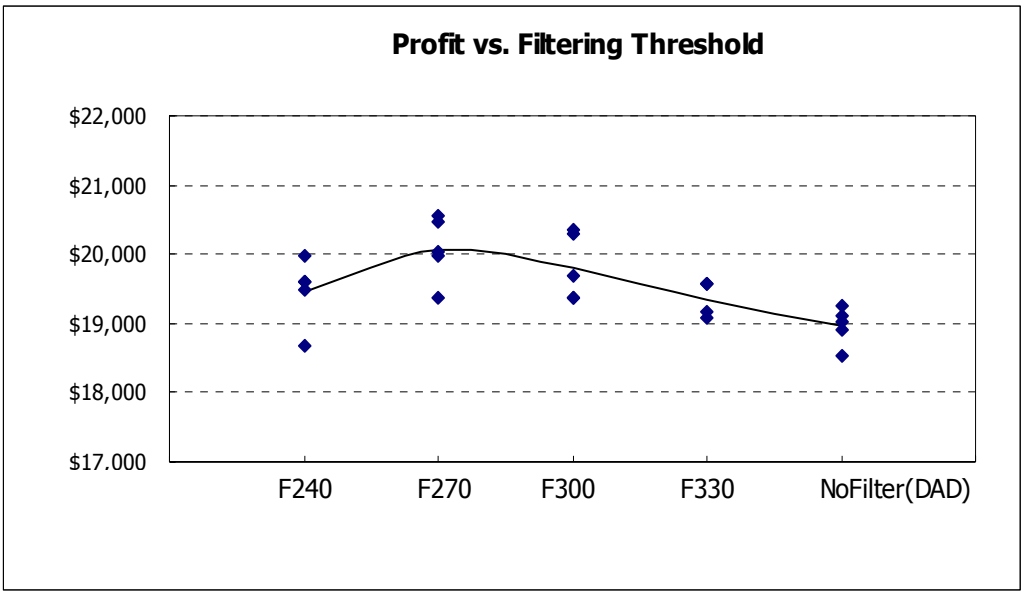


Figure 6.4 Performance results with various filtering thresholds

Figure 6.5 and table 6.2 summarize the performance of the intelligent filtering approach (*AddCost*), which utilizes additional cost information resulting from the initial assignment (*Assign I*). The performance of this process is contrasted to the performance of *DAD* without filtering and of the simple filtering approach with fixed F270 threshold. The table reports average values and standard deviations for each performance measure. In this experiment, the acceptance decision is specified as 18 miles. In order to be accepted, a new demand must satisfy two conditions. First, when the demand arrives, the system has fewer demands than 270 and second, the additional cost estimated by *Assign I* for serving this demand should be less than 18 miles.

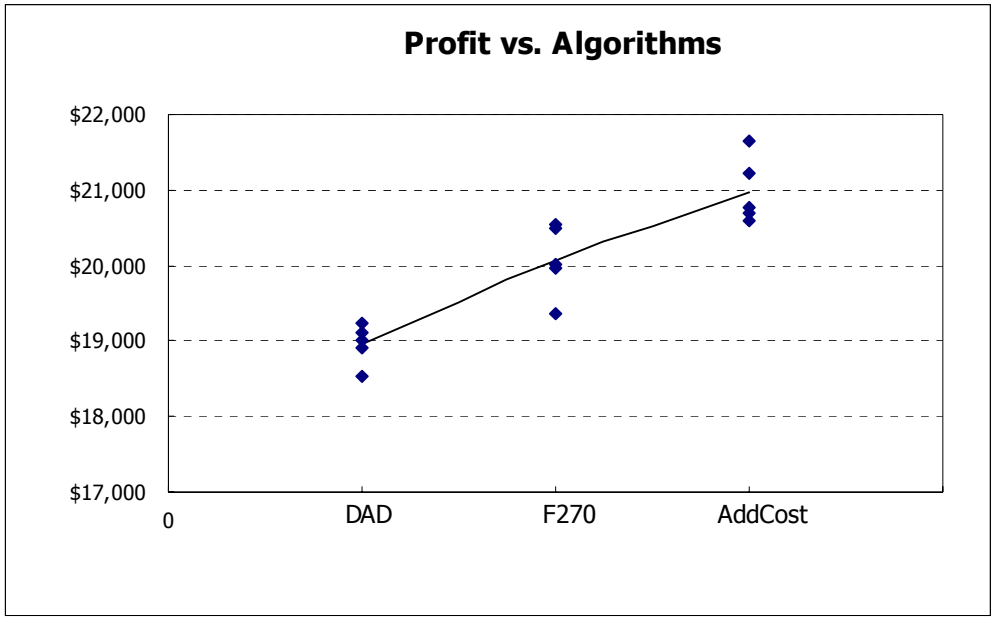


Figure 6.5 Performance results of various algorithms

Table 6.2 Performance results of various algorithms

	Profit		# of served demands		Waiting time [min]		Response time		Worst case response time [sec]
DAD	$18,964.06	*270.50*	1075.80	*14.87*	160.72	*0.69*	6.62	*2.99*	22.7
Filter 270	$20,077.76	*475.62*	1051.80	*21.78*	125.89	*0.92*	2.97	*2.88*	16.18
AddCost	$20,981.28	*436.89*	1087.20	*18.73*	121.20	*2.01*	3.04	*2.76*	18.36

The figure and table show that the additional cost filtering procedure (*AddCost*) increases the profit by 4.5% over simple filtering and 10.6% over no-filtering *DAD* algorithm. Furthermore, the response time and the waiting time are improved significantly, because the number of demands in the system affects the computation time of an assignment and the job queue length of the vehicles.

6.7 SUMMARY

This chapter presented the dynamic adaptive dispatching (*DAD*) policy for over-saturated demand condition. It applies various assignment techniques adaptively depending on the state of the system. This algorithm provides opportunities to improve a schedule by re-optimizing existing routes while keeping the response time within a tolerable range. A filtering process designed to control the number of waiting jobs in the system below 'holding capacity' was also presented. This process provides room for effective reassignment (swapping and re-sequencing), resulting in improvement in overall system efficiency. Additional improvement can be attained by combining this filtering process with

a simple intelligent acceptance decision rule, which utilizes the information obtained from *Assign I*.

Simulation experiments were conducted to assess the relative performance of the dynamic dispatching procedures and associated filtering algorithms. Results show that the *DAD* algorithm implementation process significantly improves profit and reduces the worst response time relative to the previously established benchmark though a slight increase in the average response time is observed. The filtering process also produces significant improvement in both profit and response time. These results indicate that keeping the number of waiting jobs in the queue below the 'holding' capacity (at about 75% of the 'holding' capacity in this case) is more beneficial than accepting and holding as many demands as possible when operating in an over-saturated demand environment. The intelligent filtering procedure (AddCost), improves the dispatching system efficiency further.

Chapter 7 Load Acceptance Decisions with Priority Demand

7.1 INTRODUCTION

This Chapter presents load acceptance decision policies when the carrier provides two classes of service, corresponding to various customer requirements in terms of time sensitivity vs. price sensitivity. Other problem settings are similar to those of Chapter 6. In particular, over-saturated demand situations are assumed: the demand exceeds the system's average capacity. Therefore, the dispatcher has the opportunity to select more profitable demands over less profitable ones. For these problems, the vehicle assignment (routing and scheduling) decisions follow the dynamic operating policy, *DAD* presented in Chapter 6. However, the second MIP formulation (for two demand classes) presented in Section 3.3.1 is applied to solve the local snapshot problems.

As noted, the carrier provides two types of delivery service The customers' choice between these two types of service, denoted by demand type (ξ_i), is also revealed dynamically as the routes are executed, along with other demand information including origin, destination locations, and time-window. A fraction of customers requires time sensitive 'priority' or express and on-time delivery service. In other words, they are willing to pay a premium for on-time and earlier delivery. Thus, the time-window width for this type is relatively narrow compared to that of the 'regular' type demand. Type I time-windows, defined in Section 3.1.3, are employed to represent the priority demands. The other customers are more sensitive to price, and request the 'regular' low-price

service. This class of demands has wider and flexible time-windows, represented using Type II windows. These allow pick up of a regular demand i ($\xi_i = 1$) after its critical time (τ_i^{cr}). However, a penalty is charged depending on the amount of over time $(\delta_i - \tau_i^{cr})^+$ and haul-length, l_i of the demand. The demand, however, must be picked up before the latest pickup time (τ_i^+).

When demands are classified into two types, the objective of the fleet management is to find a policy to maximize profit over the demands requested during the time horizon [0, T] as follows.

$$V''^* = \max_{\pi \in \Pi} \sum_{i}^{A_i < T} D_i^{\pi} (R_i - \beta(\varphi_i^{\pi} + l_i) - \gamma l_i \xi_i (\delta_i^{\pi} - \tau_i^{cr})^+)$$

where the reward earned from a load (R_i) depends on the type of the demand (ξ_i), in a way that different unit prices per unit-loaded distance are applied ($r_o >> r_1$). All other terms are defined previously in Section 3.2

7.2 ACCEPTANCE DECISION POLICY

7.2.1 Conceptual Framework

A general optimal acceptance rule in a dynamic and stochastic knapsack problem (Papastavrou et al., 1996) and distribution problem (Kleywegt & Papastavrou, 1998) can provide good insight into the acceptance decision for dynamic fleet management problems. This concept can be applied to this problem as follows. Suppose that, upon arrival of demand j, the current (at time $t=A_j$) locations and status of the vehicles under a certain policy π are represented by

$\Gamma_t^\pi = \{\Gamma^\pi(k,t), k = 1, ..., K\}$ with the vehicle routing schedule of the fleet, $Q_t^\pi = \{q^{k,\pi}(t), k = 1, ..., K\}$. When the demand *j* is accepted, let $\overline{Q}_t^\pi$ represent the updated routing schedule including demand *j* under policy π at time *t*. Note that the locations and status of vehicles do not change. In other words, it is assumed that the solution execution time is ignored. With these notations, the optimal acceptance rule is defined as follows:

$$
D^*(\Gamma_t^\pi, Q_t^\pi, j)
$$

$$
= \begin{cases}
1 & \text{if demand } j \text{ is feasible} \\
 & \text{and } R_j > E\Big[\sum_{i}^{A_j < A_i < T} D_i^\pi (R_i - \beta(\varphi_i^\pi + l_i) - \gamma \xi_i l_i (\delta_i^\pi - \tau_i)^+) \Big| \Gamma_t^\pi, Q_t^\pi\Big] - \\
 & \qquad E\Big[\sum_{i}^{A_j < A_i < T} D_i^\pi (R_i - \beta(\varphi_i^\pi + l_i) - \gamma \xi_i l_i (\delta_i^\pi - \tau_i)^+) \Big| \Gamma_t^\pi, \overline{Q}_t^\pi\Big] \\
0 & \text{if demand } j \text{ is not feasible} \\
 & \text{or } R_j \le E\Big[\sum_{i}^{A_j < A_i < T} D_i^\pi (R_i - \beta(\varphi_i^\pi + l_i) - \gamma \xi_i l_i (\delta_i^\pi - \tau_i)^+) \Big| \Gamma_t^\pi, Q_t^\pi\Big] - \\
 & \qquad E\Big[\sum_{i}^{A_j < A_i \le T} D_i^\pi (R_i - \beta(\varphi_i^\pi + l_i) - \gamma \xi_i l_i (\delta_i^\pi - \tau_i)^+) \Big| \Gamma_t^\pi, \overline{Q}_t^\pi\Big]
\end{cases}
$$

If load *j* is accepted, then expected total profit is defined as follows. The expected profit is estimated over the demands that will be requested during the time horizon (A_j, T) not counting load *j*, given state of the system represented by vehicles' status and locations as well as the updated routing schedule including demand *j*:

$$E[\sum_{i}^{A_j < A_i \le T} D_i^{\pi}(R_i - \beta(\varphi_i^{\pi} + l_i) - \gamma\xi_i l_i(\delta_i^{\pi} - \tau_i)^+)\Big|\Gamma_t^{\pi}, \overline{Q}_t^{\pi}]$$

Otherwise, if the load is rejected, in this case the state of the system is identical to the state just before the demand comes in, and the expected total profit is:

$$E[\sum_{i}^{A_j < A_i \le T} D_i^{\pi}(R_i - \beta(\varphi_i^{\pi} + l_i) - \gamma\xi_i l_i(\delta_i^{\pi} - \tau_i)^+)\Big|\Gamma_t^{\pi}, Q_t^{\pi}]$$

Hence, the expected marginal total profit (*EMP*) by rejecting the load *j* is:

$$EMP(j) = E[\sum_{i}^{A_j < A_i \le T} D_i^{\pi}(R_i - \beta(\varphi_i^{\pi} + l_i) - \gamma\xi_i l_i(\delta_i^{\pi} - \tau_i)^+)\Big|\Gamma_t^{\pi}, Q_t^{\pi}]$$
$$- E[\sum_{i}^{A_j < A_i \le T} D_i^{\pi}(R_i - \beta(\varphi_i^{\pi} + l_i) - \gamma\xi_i l_i(\delta_i^{\pi} - \tau_i)^+)\Big|\Gamma_t^{\pi}, \overline{Q}_t^{\pi}]$$

The *EMP(j)* represents the expected marginal profit that the capacity of the system, which is held by rejecting demand *j*, would produce by serving future demands. In other words, the carrier may lose the expected profit by accepting load *j,* because accepting demand *j* may preclude the system from accepting future demands that would generate more profit. If the system has a few loads in the queue so that the system has enough room for accepting future demands, the expected marginal profit by rejecting demand *j* is negligible. In contrast, as the system is saturated with the demands that have been accepted but not-yet-picked

up, the expected marginal profit by rejecting demand j tends to increase. Therefore, the proper acceptance decision criteria for demand j is as follows: At first, the load j should be feasible. In addition, the revenue earned from accepting the load j should be greater than the expected marginal profit by rejecting demand j.

This acceptance decision rule makes intuitive sense. However, it is extremely difficult to estimate the expected total profit over the future (unrealized) demands at a given system state (which is described by locations and status of vehicles, along with associated scheduled demands). Furthermore, even if it were possible to estimate the expected profit, the computation time of the estimation process would be restricted under this dynamic operating environment since the acceptance decision should be communicated to the customer in a short time after the request is made. Therefore, the acceptance decision policy for this dynamic fleet management problem should be able to reach a decision in a short time, and should take into account the system state upon arrival of a load.

In the problem of interest, customers can choose the service type out of two mutually exclusive alternatives. The priority service is more expensive than the regular service in a way that the price rate (unit price per unit distance) of a priority demand (r_o) is much higher than the regular demand's unit price (r_1). Thus, the dispatcher favors priority demands. Furthermore, although a long-haul regular demand could generate the same revenue as a short-haul priority demand, the priority load is more valuable to the carrier because it consumes less of the fleet's resources. Short haul-length corresponds to less operating cost, and the

short service (delivery) time provides opportunity for the driver to serve additional demands.

Therefore, the basic approach to maximize the overall profit is to accept as many priority demands as possible while controlling the acceptance decision of the regular demands. This control process aims to manage the state of the system, under which the carrier can accept future priority demands without the risk of underutilizing the transportation resources. In addition, this acceptance decision policy, as reported in Chapter 6, may improve the efficiency of the reassignment procedure by keeping the total number of demands in the system below the 'holding capacity'.

The ability to serve a given demand depends on the demand's time-window as well as its relative location vis a vis the locations of the fleet of vehicles in the system. Both of these characteristics are not typically known a priori, therefore it is difficult to guarantee that a future (unrealized) priority demand will be served, even if a present regular demand is rejected. For this reason, the system ability to accept a (unrealized) priority demand is expressed in probabilistic terms rather than a number of discrete slots.

7.2.2 Feasibility Index

This section presents the 'Feasibility Index' (*FI*), which represents a system state upon arrival of a regular demand in terms of the approximate expected number of vehicles that can serve an unrealized (future) priority demand. Due to the complexity of the probability estimation (the probability that

the system can accept a future priority demand), the approximation algorithm is developed based on several assumptions.

First, it is assumed that the time-window width of a priority demand is known in advance. This assumption is applicable because the service provider (carrier) can pre-specify the delivery service characteristics.

Second, the arrival time of a future priority demand is assumed to be the same as the time that *FI* is estimated. This would correspond to a priority demand being requested immediately after arrival of a regular demand. The justification for this assumption is that until the next demand comes in, the ability of the system to accept a future priority demand improves gradually in the course of time, since the vehicles are serving the accepted demands on a continuous basis. Thus, *FI* provides a lower bound for the expected number of vehicles that can accept a future priority demand.

Third, the estimation process of the probability employs an insertion heuristic method when searching feasible routing schedules for a future demand. Note that the feasibility of a demand should be assured by a feasible routing schedule including the demand. Even though this method cannot explore all possible routing schedules for a future demand, it is guaranteed that once a feasible schedule could be constructed with this method, the initial assignment procedure would be able to accept the demand and construct a feasible routing schedule.

Finally, the *FI* estimation process involves probabilistic information about haul-length of a priority demand and the required empty distance to serve it.

The probability density function of the haul-length can be obtained from the historical data. Furthermore, a threshold (**β**) is pre-specified representing the maximum allowable empty distance required to serve a demand. The empty distance value is essential when estimating *FI* using the insertion heuristic method. The longest possible empty distance, in this problem setting 141.4 miles, can be applied to provide a lower bound. This value, however, corresponds to an extremely rare case, in which all vehicles are located at one corner of the geographic region under consideration and the new demand's origin location is the opposite corner. The likelihood of this event is negligible. Furthermore, the *FI* value estimated using the largest value significantly underestimates the expected number of vehicles that can serve a future priority demand. Therefore, the average plus one standard deviation of the empty distance required to serve a demand (obtained from the historical data) would be used as the threshold (**β**).

The estimation process based on the assumptions stated above is as follows.

$$FI(k) = FI(k \mid \Gamma^{\pi}(k,t), q^{k,\pi}(t))$$

$$= \begin{cases} if\ s^{k,\pi} = 1\ (loaded) \\ \quad max(\{FI(k,(q_i^{k,\pi}, q_{i+1}^{k,\pi})),\ for\ i = 1,\ldots,I-1\},\ FI(k, q_I^{k,\pi})) \\ if\ s^{k,\pi} = 2\ (empty) \\ \quad max(FI(k, q_1^{k,\pi}), \{FI(k,(q_i^{k,\pi}, q_{i+1}^{k,\pi})),\ for\ i = 1,\ldots,I-1\},\ FI(k, q_I^{k,\pi})) \\ if\ s^{k,\pi} = 3\ (idle) \\ \quad FI(k) \end{cases}$$

where **I** represents queue length ($I = |q^{k,\pi}(t)|$), and for notational simplicity current time **t** is omitted and $FI(k,(q_i^{k,\pi}, q_{i+1}^{k,\pi})) = FI^T(k, (q_i^{k,\pi}, q_{i+1}^{k,\pi}))^* FI^L(k, (q_i^{k,\pi}, q_{i+1}^{k,\pi}))$

$$FI(t \mid \Gamma_t^\pi, Q_t^\pi) = \sum_{k=1}^{k=K} FI(k \mid \Gamma^\pi(k,t), q^{k,\pi}(t))$$

The first step is to estimate, *FI(k),* the probability that single vehicle *k* can accept a future priority demand, in which the current location, status ($\Gamma^\pi(k,t)$) of vehicle k and associated scheduled demands ($q^{k,\pi}(t)$) are known to the dispatcher. This process is based on the given system state. There are two main features of the system that affect the *FI(k)*. The first feature is the updated location of vehicle *k* considering the vehicle status. The other feature is the set of times, $\{(\tau_i^+ - t), i = \{q^{k,\pi}(t)\}\}$, where *t* represents the decision epoch. This set of times is obtained from the latest pickup times (τ_i^+) of the demands, which are scheduled in vehicle *k*'s queue ($i = \{q^{k,\pi}(t)\}$). Considering those features, *FI(k)* estimation process begins with the location-based Feasibility Index of vehicle *k* ($FI^L(k)$).

Suppose that an idle vehicle locates in a square region, and the time-window width of a priority demand is known in advance as τ ($\tau_i^+ - \tau_i^-$). In this case, the vehicle can serve a requested demand only if the vehicle can reach the origin of the requested demand within τ. In addition, a pre-specified threshold β is applied to limit the maximum empty movement of the vehicle to serve the demand. Thus, $\tau' = \min(\tau, \beta)$ is a updated threshold. In other words, when vehicle *k* is idle the $FI^L(k)$ is the probability that origin of a priority demand is generated in the area specified by τ'**.** For example, if an idle vehicle is located as shown in the

Figure 7.1, the vehicle can serve a future demand only if a future demand is requested the origin of a future demand is in the overlapped shaded are. If it is assumed that origin location of a demand is randomly distributed over the square region, the location-based Feasibility Index of vehicle *k*, $FI^L(k)$ is represented by the size of the shaded area over the size of the rectangular area.

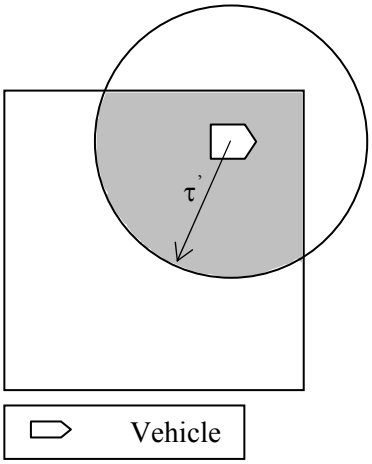


Figure 7.1 $FI^L(k)$ for an idle vehicle

If a vehicle is not idle, the location and $FI^L(k)$ of the vehicle should be updated considering the routing schedule of the vehicle. For example, when a vehicle (with only one demand in its queue) is loaded status, namely on the way to the destination of a load, the vehicle will be available at the destination of the current load. Therefore updated τ' should be reduced taking into account available time of the vehicle and $FI^L(k)$ should also accommodate the update location of vehicle *k*.

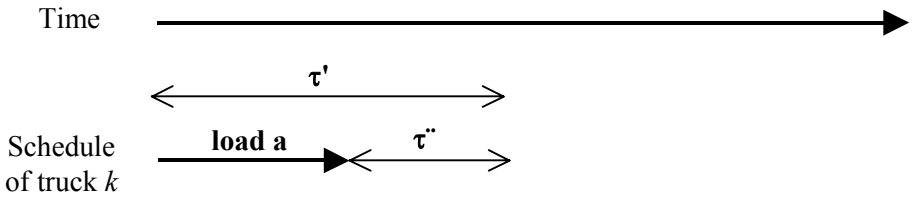


Figure 7.2 Update process of τ' to $\tau^{\cdot\cdot}$ when vehicle *k* is loaded status with one demand, load **a**, in it's queue

Figure 7.2 depicts the reduction of τ, where solid line represents the loaded movement. Of course, the threshold β is also considered. Note that, calculation of the $FI^L(k)$ assumes that a new priority demand with time-window width τ comes into the system immediately after current decision process. This value provides the lower bound of τ', since as the next demand arrival time is late, the value of the τ' increases.

If a vehicle has more than two demands in its schedule (or one demand and empty status), the probability that a priority demand can be inserted into existing service schedule should be taken into consideration along with the location based estimation process. In this case, the time-windows (particularly the latest times) of the existing demands in the queue (τ_i^+, $i = \{q^{k,\pi}(t)\}$) are important factors that affect the Feasibility Index. This type of *FI* is called time-window based Feasibility Index ($FI^T(k)$). For example, a vehicle has scheduled three loads (**a, b, c**) sequentially as shown in Figure 7.3. The possible insertion

slots for a new demand are (**a, b**), (**b, c**) and end of the queue, where (**a, b**) denotes the slot between demands **a** and **b**.

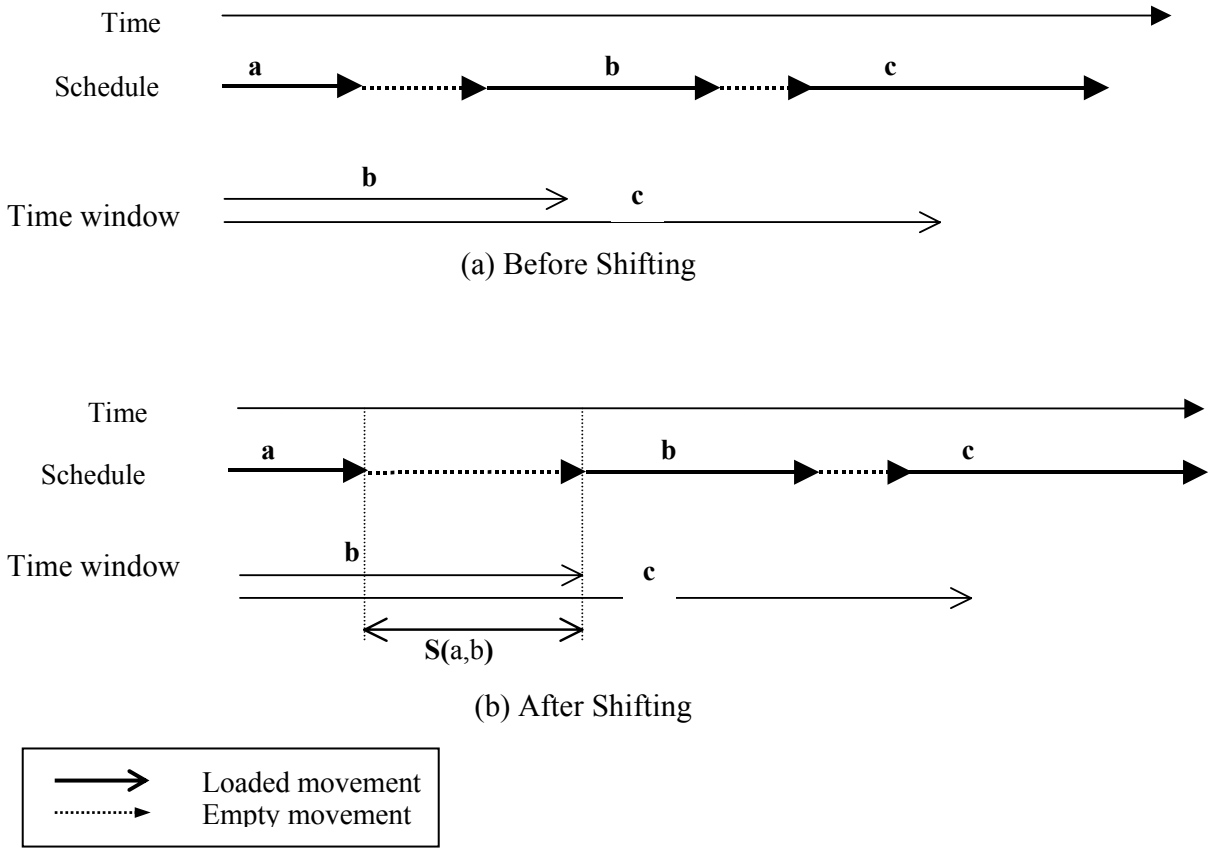


Figure 7.3 Maximum allowable time-space between loads **a** and **b**

The maximum time-space room between loads **a** and **b** can be obtained by shifting the scheduled pick up times of load **b** and **c** up to the points that their time-windows allow. Let **S (a,b)** be the maximum allowable time-space between demand **a** and **b.**

$$\mathbf{S}(\mathbf{a},\mathbf{b}) = TO(\mathbf{b}) - TD(\mathbf{a}) + \min\{(\ \tau_b^+ - TO(\mathbf{b})),\ (\ \tau_c^+ - TO(\mathbf{c}))\}$$

TO(j): scheduled time at the origin of load j

TD(j): scheduled time at the destination of load j

In order to insert a demand between demand **a** and **b**, the maximum time-space **S**(**a**,**b**) should be greater than the haul-length of the (future, unrealized) demand plus required empty distance to serve it. For the haul-length, a random variable (η) is defined following a probability density function (*pdf*) obtained from the historical data. With respect to the required empty distance to serve the demand, average plus one standard deviation value from the historical data would be used as the maximum allowable empty distance (**β)**.

$$FI^{T}(k, (a,b)) = Pr\{S(a,b) > \eta + \beta\}$$

Where $FI^{T}(k, (a,b))$ denotes the probability that a priority demand can assigned between demands **a** and **b**. In addition to this time-window related probability, the previously presented location based probability $FI^{L}(k, (a,b))$ is also taken into account with updated **τ** considering updated vehicle location (in this case, at the destination of the load **a**).

$$FI(k, (a,b)) = FI^{T}(k, (a,b)) * FI^{L}(k, (a,b))$$

This procedure is applied into every available slot including the case of appending the new demand at the end of the queue, $FI^{L}(k, c)$ such that the

vehicle's updated location is the destination of demand **c.** Calculation process of $FI^L(k, c)$ regards the demands **a**, **b** and the empty movement between the two demands as a single long demand and consider updated location of the vehicle. Thus, the feasibility index of vehicle *k*, *FI(k)* is as follows.

$$FI(k) = max \{ FI(k, (a,b)), FI(k, (b, c)), FI(k, c)\}$$

This estimation procedure explores all available slots of vehicle *k*, to which a new demand can be inserted depending on the vehicle status. Note that if a vehicle is empty moving status, the slot between the vehicle and the first demand should be considered. This slot represents the en-route diversion case.

Finally, feasibility index of the system is the sum of *FI(k)'s* for all vehicles in the fleet, which represents the expected number of vehicles that can accept a future priority demand with empty movements less than **β.**

7.2.3 *FI* Based Acceptance Decision Policy

Whenever a demand arrives, the feasibility check process, *Assign I* presented in Chapter 6, is applied. If the demand is feasible and is a priority demand, it is accepted outright. Otherwise, if the demand is a regular demand, initial routing schedule involving the new load is constructed. Then, *FI* is estimated based on this updated state of the system. Only if the $FI(t \mid \Gamma_t^\pi, \overline{Q_t^\pi})$ is greater than specified threshold (FI^*), the requested demand is accepted.

Otherwise, if the $FI(t \mid \Gamma_t^{\pi}, \overline{Q_t^{\pi}})$ is less than FI^*, the load is rejected in order to increase *FI*.

$$D^*(\Gamma_t^{\pi}, Q_t^{\pi}, j) = \begin{cases} 1 & \text{if } \xi_j = 0, \text{ and demand } j \text{ is feasible} \\ & \text{or} \\ & \xi_j = 1, \text{ and demand } j \text{ is feasible, and } FI(t \mid \Gamma_t^{\pi}, \overline{Q}_t^{\pi}) \geq FI^* \\ 0 & \text{if demand } j \text{ is not feasible} \\ & \text{or} \\ & FI(t \mid \Gamma_t^{\pi}, \overline{Q}_t^{\pi}) < FI^* \end{cases}$$

7.3 NUMERICAL RESULTS

The simulation experiment designed in this chapter is similar to the simulation framework of the Chapter 6. The only difference is the newly introduced time-window types. The type I time-windows are employed for the priority demands with fixed 1.5 hour length (τ_i^-, τ_i^+) to characterize the express and on-time delivery service. The regular demand employs the Type II time-window with total 4 hours duration (τ_i^-, τ_i^+), of which one hour is reserved for over time (τ_i^{cr}, τ_i^+). If the pick up of regular demand *i* occurs after its critical time ($\delta_i > \tau_i^{cr}$), then a penalty in proportion to ($\delta_i - \tau_i^{cr}$) and the haul-length (l_i) is charged. The maximum penalty is charged when the demand is picked up at the latest pickup time of the demand, $\delta_i = \tau_i^+$. In this case, 20% of the revenue earned

from demand i is the maximum penalty, which is specified by the scaling parameter, γ.

Two scenarios are tested with different distribution ratios between two demand types. In the first scenario, 6.25% of the requested demands are the priority demands. On the other hand, the second scenario evaluates the case that 25% of the requested demands are the priority demands. The experiments simulate 5 days operating of the fleet where 1265.3 (±14.2) demands including both types are requested in a day, and 77.3(±4.8) and 310.3 (±11.2) demands are priority demands, respectively.

Two benchmark policies are tested to evaluate the performance of the *FI* acceptance decision policy. The first benchmark policy accepts as many demands as possible, regardless of the demand type i.e., feasibility-based-acceptance rule. The second benchmark is the simple filtering process presented in Chapter 6.

To find the optimal FI^* threshold value, calibration process with various FI^* values are conducted as like in Chapter 6. Figure 7.4 illustrates the results with various FI^* values in both scenarios. The optimal threshold values showing best performance are 7 and 6 for 25% and 6.25% of priority demand portions, respectively.

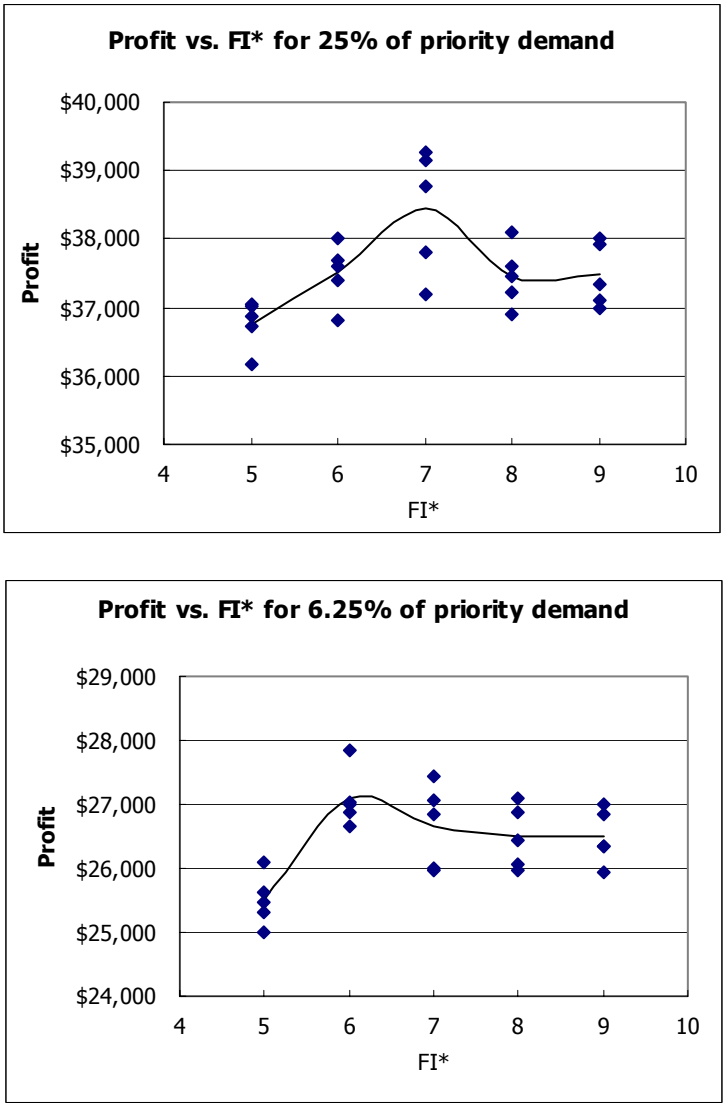


Figure 7.4 Performance with various FI^* thresholds

Table 7.1 shows the numerical results with *FI** values for the two scenarios. As seen in the table, in terms of the total profit (column 5), the performance of the dispatching system significantly improves with the proposed

FI acceptance decision policy compared to both of the benchmarks in both scenarios. The sixth and the last columns present the total number of served demands and served priority demands respectively along with the acceptance ratios. In spite of the similar acceptance ratio with respect to the total number of accepted demands, the more accepted priority demands improves the total profit significantly. For example, in the 25% priority demand scenario, although the FI-based acceptance decision policy accepts less total number of demands (850.6, 67.2%) compared to feasibility-based acceptance decision policy (900.2, 71.1%), the high acceptance ratio of the priority demand (86.1%) produces much greater overall profit ($38,438) than the feasibility-based acceptance decision policy case ($23,307). Furthermore, it is notable that the over time cost (column 3) is successfully reduced by controlling the number of demands in the system with filtering process and FI based acceptance decision policy.

Table 7.1 Numerical results of the FI based acceptance decision policy

(a) 6.25% of demands are priority demand

Acceptance decision policy	Total Revenue (std.)	Total Cost (std.)	Over Time Cost (std.)	Total Profit (std.)	Total # of Served Demands (std.)	# of served Priority demands (std.)
Feasibility based acceptance	$56,663	$32,312	$5,340	$19,011	896.8 (70.9%)	25.8 (33.4%)
	(618)	(229)	(363)	(549)	(13.2)	(3)
Simple Filtering	$57,295	$31,937	$1,249	$24,109	920.8 (72.8%)	35.2 (45.6%)
	(412)	(449)	(205)	(484)	(10.1)	(5)
FI	$59,879	$31,820	$979.2	$27,080	897.2 (70.9%)	66.3 (85.6%)
	(460)	(184)	(168)	(453)	(11.2)	(7)

(b) 25% of demands are Priority demands

Acceptance decision policy	Total Revenue (std.)	Total Cost (std.)	Over Time Cost (std.)	Total Profit (std.)	Total # of Served Demands (std.)	# of served Priority demands (std.)
Feasibility based acceptance	$60,712	$32,485	$4,920	$23,307	900.2 (71.1%)	102.8 (33.1%)
	(572)	(223)	(232)	(492)	(6.3)	(4.5)
Simple Filtering	$61,622	$31,483	$1,267	$28,872	903.8 (71.4%)	139.8 (45.1%)
	(910)	(655)	(302)	(335)	(16.5)	(7.7)
FI	$68,829	$30,038	$352	$38,438	850.6 (67.2%)	267.2 (86.1%)
	(1090)	(535)	(98)	(804)	(13.2)	(11.2)

The evolution of the total number of demands in the system over time within a day is shown in Figure 7.5. For the presentational simplicity, only one day's performance of the high portion of priority demands (25%) scenario is presented, the shaded line represents the total number of demands in the system over time with the feasibility-based-acceptance policy, the solid line with the simple filtering, then bolded line with the *FI*-based acceptance policy. After the 8^{th} hour, the company does not accept any new demands, and focuses on serving the demands that has been accepted. The solid line shows the effect of the filtering process with a threshold of 250 demands, so that all demands received when the system is at this level are rejected outright until 6.5^{th} hour. With respect to the *FI*-based acceptance decision policy, as the portion of priority demands out of the total demands in the system increase, *FI* value at time *t* decreases due to the decrease of the average time-window width over the demands in the system. Therefore, in order to maintain the *FI(t)* value at a certain level, the total number of demands in the system decreases by rejecting regular demands. Note that, these acceptance decision policies were applied until the 6.5^{th} hour in order to hold as many demands as possible at the 8^{th} hour.

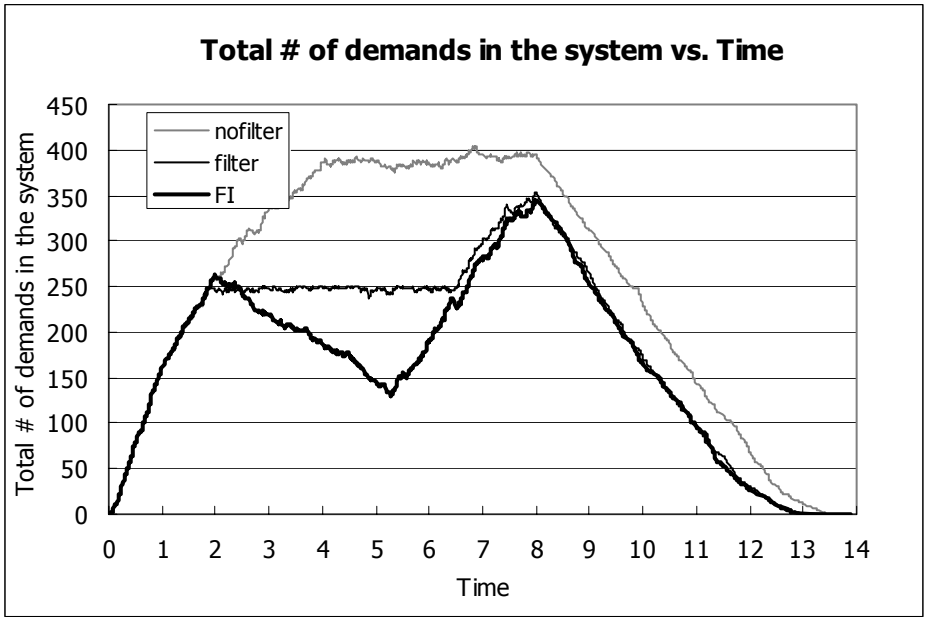


Figure 7.5 Total number of demands in the system with various acceptance decision policies

7.4 SUMMARY

In this chapter, two classes of service are introduced corresponding to various customer requirements. For these two types of demand, the Feasibility Index based dynamic acceptance decision policy is proposed. The *FI* represents the approximation of the expected number of vehicles being able to serve future (still unrealized) priority demand based on the current system state, in which the maximum required empty distance parameter (β) is pre-specified. Simulation experiment results show that the *FI* acceptance decision policy significantly improves the total profit, particularly increase the portion of accepted priority

demand. Furthermore, this policy significantly reduces the delay cost with respect to the regular demand service.

Chapter 8 Conclusion

This Chapter reviews the objectives of the thesis outlined in Section 1.4, and presents related conclusions. Furthermore, the main contribution of this research is addressed along with the recommended future work. The first objective was achieved by providing a formal definition of the dynamic fleet management problem taking advantage of real-time information on vehicle locations and states, as well as by discussions about various problem features. Secondly, two MIP formulations are developed based on Yang, Jaillet, and Mahmassani (2000, 2002)'s formulation for the local snapshot problems (deterministic and static truckload multi-vehicle routing problems for pickup-and-delivery service with time-windows) depending on demand characteristics. The first formulation deals with homogeneous demands, while the second formulation models the problem with two types of demand corresponding various customer requirements. These MIP models are applied in the proposed dynamic hybrid decision policies as needed. As stated in the third objective, variations of the hybrid dynamic decision policies are developed according to the problem settings. Those policies successfully satisfy the dynamic operational requirements: quick (with small variance) response to a customer with respect to the acceptance decision and full utilization of the computational resources without wasting available time between successive demand requests. In addition, developed partitioning strategies provide a solution approaches to a problem managing a large fleet of vehicles. Regarding to the real-time acceptance/ rejection decision

stated in the fourth objective, various decision policies are developed corresponding to a range of demand situations. Major findings include that controlling the number of demands in the system under the 'holding capacity' improves overall system efficiency. In addition, a policy for the acceptance decision on two types of demands is developed and evaluated. The final objective was accomplished by developing simulation frameworks to evaluate the effectiveness of policies developed. Furthermore, a simple local heuristic approach (Regan, Mahmassani & Jaillet, 1996a) and *RAPID-SL* (Powell, Snow, & Cheung, 2000) algorithm are coded and implemented in this simulation framework as benchmark policies and then the comparative results are reported.

The main contribution of this research is to provide solution algorithms for local snapshot version of the dynamic fleet management problems and dynamic operational policies using those algorithms. The nature of these problems incorporate large amount of uncertainties mainly due to the dynamically requested demands, and the algorithm execution time to solve the problems is one of the main obstacles to direct application of the solution algorithms due to the complexity of the problem as well as the dynamic operational restrictions. Various efficient solution algorithms are developed for the static and deterministic version of the problems (local snapshot problems) with the aim of solving the problems as close to optimality as possible with minimum computing time. With adaptive application of these algorithms, a variety of dynamic policies are then developed to provide methodological approaches fully utilizing the available hardware and computational resources while keeping the service quality, as well

as to explicitly recognize their computational characteristics for implementation in a dynamic operational setting. The insight from this work can be expanded to wide range of dynamic and stochastic fleet management problems.

The recommended future research would consider multiple pickups and deliveries for less-than truckload delivery service considering the truck capacity constraints. Moreover, explicit consideration of variable travel time of the vehicles due to unpredictable events such as network congestion and/or accidents, are also recommended for future study. Particularly in urban area, travel time between two points varies significantly and affects the system performance. Nevertheless, most of the routing and scheduling algorithms utilize distance rather than actual travel time. Therefore, combining dynamic fleet management system with the system being able to provide more reliable forecasted (when the scheduled vehicle movements are actually realized) travel time such as 'Dynamic Traffic Assignment' system would be recommended.

References

Abdelwahab, W.M. & Sargious, M.A. 1991, 'A Simultaneous Decision-making Approach to Model the Demand for Freight Transportation', *Canadian J. of Civil Eng.*, vol.18, pp. 515-520.

Assad, A.A. 1988, 'Modeling and Implementation Issues in Routing', in *Vehicle Routing: Methods and Studies,* eds. B.L. Golden & A.A. Assad, Elsevier Science Publishers, Amsterdam, pp.249-291.

Badeau, P., Guertin, F., Gendreau, M. & Potvin, J.-Y. 1997, 'A Parallel Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows', *Transportation Research Part C*, vol.5, no.2, pp. 109

Barnhart, C. & Ratliff, H.D. 1993, 'Modeling Intermodal Routing', *Journal of Business Logistics*, vol.14, no.1, pp. 205-223.

Bartholdi, I.J.J. & Platzman, L.K. 1988, 'Heuristics Based on Spacefilling Curves for Combinatorial Problems in Euclidean Space', *Management Science*, vol.34, pp. 291-305.

Bertsimas, D.J., Jaillet, P. & Odoni, A.R. 1990, 'A Priori Optimization', *Operations Research*, vol.38, no.6, pp. 1019-1033.

Bertsimas, D.J. & Van Ryzin, G. 1993, 'Stochastic and Dynamic Vehicle Routing in the Euclidean Plane with Multiple Capacitated Vehicles', *Operations Research*, vol.41, no.1, pp. 60-76.

Bertsimas, D.J. & Van Ryzin, G. 1991, 'A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane', *Operations Research*, vol.39, no.4, pp. 601-615.

Bixby, R.E. & Lee, E.K. 1998, 'Solving a Truck Dispatching Scheduling Problem Using Branch-and-Cut', *Operations Research*, vol.46, no.3, pp. 355-367.

Calvo, R.W. 2000, 'A New Heuristic for the Traveling Salesman Problem wiht Time Windows', vol.34, no.1, pp. 113-124.

Cardell, N., Huang, S. & Brown, S. 1995, 'Decision making in the Motor Carrier Industry', *Transportation Research Part A*, vol.29A, no.6, pp. 401-419.

Cheung, R. & Muralidharan, B. 2000, 'Dynamic Routing for Priority Shipments in LTL Service Networks', *Transportation Science*, vol.34, no.1, pp. 86-98.

Chiang, W.-C. & Russell, A. 1996, 'Simulated Annealing Metaheuristics for the Vehicle Routing Problem with Time Windows.', *Annals of Operations Research*, vol.63, no.3, pp. 3-27.

Christofides, N. 1985, 'Motivation and modeling', in *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization,* eds. E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan & D. Shmoyes, John Wiley and Sons., New York

Dejax, P.J. & Crainic, T.G. 1987, 'A Review of Empty Flows and Fleet Management Models in Freight Transportation', *Transportation Science*, vol.21, no.4, pp. 227-247.

Eglese, R.W. 1990, 'Simulated Annealing: A Tool for Operational Research', *EJOR*, vol.46, pp. 271-281.

Fisher, M. 1995, 'Vehicle Routing', in *Network Routing,* eds. M.O. Ball, T.L. Magnanti, C.L. Monma & G.L. Nemhauser, Elsevier Science, Amsterdam

Frantzeskakis, L. & Powell, W.B. 1990, 'A Successive Linear Approximation Procedure for Stochastic, Dynamic Vehicle Allocation Problem', *Transportation Science*, vol.24, no.1, pp. 40-57.

Gans, N. & Van Ryzin, G. 1999, 'Dynamic Vehicle Dispatching: Optimal Heavy Traffic Performance and Practical Insight', *Operations Research*, vol.47, no.5, pp. 675-692.

Garfinkel 1985, 'Vehicle Routing', in *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization,* eds. E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan & D. Shmoyes, John Wiley and Sons, New York

Garrido, R.A. 1997, Analysis of Spatial Temporal Characteristics of Freight Demand. Ph.D. Thesis, The University of Texas at Austin.

Garrido, R.A. & Mahmassani, H.S. 1998, 'Forecasting Short-Term Freight Transportation Demand Poisson STARMA Model', *Transportation Research Record*, vol.1645, pp. 8-16.

Gendreau, M., Guertin, F., Potvin, J.-Y. & Taillard, E. 1999, 'Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching', *Transportation Science*, vol.33, no.4, pp. 381-390.

Golden, B.L. & Assad, A.A. 1986, 'Perspectives on Vehicle Routing: Exciting New Developments', *Operations Research*, vol.34, no.5, pp. 803-810.

Higgins on, J.K. & Bookbinder, J.H. 1995, 'Markovian Decision Processes in Shipment Consolidation', *Transportation Science*, vol.29, no.3, pp. 242-255.

Hoffman, A.J. & Wolfe, P. 1985, 'History', in *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization,* eds. E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan & D. Shmoyes, John Wiley and Sons, New York

Hu, T.-Y. 2001, 'Evaluation for Dynamic Vehicle Routing Strategies Under Real-Time Information'*, TRB 80th Annual Meeting,* Washington D.C.

Ichoua, S., Gendreau, M. & Potvin, J.-Y. 2000, 'Diversion Issues in Real-Time Vehicle Dispatching', *Transportation Science*, vol.34, no.4, pp. 426-438.

Jaillet, P. & Odoni, A.R. 1988, 'The Probabilistic Vehicle Routing Problems', in *Vehicle Routing: Methods and Studies,* eds. B.L. Golden & A.A. Assad, Elsevier Science Publishers, Amsterdam, pp.293-318.

Jung, S. & Haghani, A. 2000, 'Genetic Algorithm for a Pickup and Delivery Problem with Time Windows', vol.1733, pp. 1-7.

Kim, Y., Mahmassani, H.S. & Jaillet, P. 2002, 'Dynamic Truckload Truck Routing and Scheduling in Over-Saturated Demand Situation', *Transportation Research Record*, vol.1783, pp. 66-71

Kim, Y., Mahmassani, H.S. & Jaillet, P. 2003, 'Two-Phase Optimization Approaches to Solve Large Fleet Dynamic Routing and Scheduling Problems', W*orking paper*

Kim, Y., Mahmassani, H.S. & Jaillet, P. 2003, 'Dynamic Truckload Truck Routing and Scheduling for Large Fleet with Priority Demands', W*orking paper*

Koskosidis, Y.A. & Powell, W.B. 1996, 'An Optimization-Based Heuristic for Vehicle Routing and Scheduling with Soft Time Window Constraints', *Transportation Science*, vol.26, no.2, pp. 69-85.

Laguna, M. & Marti, R. 1999, 'GRASP and Path Relinking for 2-Layer Straight Line Crossing Minimization', *Journal on Computing*, vol.11, no.1, pp. 44-52.

Larson, R. & Odoni, A.R. 1981, *Urban Operations Research*, Princeton Hall, New Jersey.

Law, A.M. & Kelton W.D. 1991, *Simulation Modeling & Analysis*, McGraw, New York.

Mahmassani, H.S., Kim, Y. & Jaillet, P. 2000, 'Local Optimization Approaches to Solve Dynamic Commercial Fleet Management Problems', *Transportation Research Record*, vol.1733, pp. 71-79.

McGeoch, C. 1996, 'Toward an Experimental Method for Algorithm Simulation', *Journal on Computing*, vol.8, no.1, pp. 1-15.

McGill, J.I. 1989, Optimization and Estimation in Airline Yield Management. Ph.D. Thesis, The University of British Columbia.

McGill, J.I. & Van Ryzin, G. 1999, 'Revenue Management: Research Overview and Prospects', *Transportation Science*, vol.33, no.2, pp. 233-256.

Nemhauser, G.L. & Wolsey, L.A. 1988, *Integer and Combinatorial Optimization*, John Wiley & Sons, New York.

Powell, W.B. 1986a, 'A Local Improvement Heuristic for the Design of Less-than-Truckload Motor Carrier Networks', vol.20, no.4, pp. 246-257.

Powell, W.B. 1986b, 'A Stochastic Model of the Dynamic Vehicle Allocation Problem', vol.20, no.2, pp. 117-129.

Powell, W.B. 1987, 'An Operational Planning Model for the Dynamic Vehicle Allocation Problem with Uncertain Demands', *Transportation Research Part B*, vol.21B, no.3, pp. 217-232.

Powell, W.B. 1988, 'Comparative Review of Alternative Algorithms for the Dynamic Vehicle allocation problem', in *Vehicle Routing: Methods and*

Studies, eds. B.L. Golden & A.A. Assad, Elsevier Science Publishers, Amsterdam, pp.249-291.

Powell, W.B. & Frantzeskakis, L. 1994, 'Restricted Recourse Strategies for Dynamic Networks with Random Arc Capacities', *Transportation Science*, vol.28, no.1, pp. 3-23.

Powell, W.B., Jaillet, P. & Odoni, A.R. 1995, 'Stochastic and Dynamic Networks and Routing', in *Network Routing,* eds. M.O. Ball, T.L. Magnanti, C.L. Monma & G.L. Nemhauser, Elsevier, Amsterdam

Powell, W.B. 1996, 'A Stochastic Formulation of the Dynamic Assignment Problem, with an Application to Truckload Motor Carriers', *Transportation Science*, vol.30, no.3, pp. 195-218.

Powell, W.B. & Carvalho, T.A. 1998a, 'Dynamic Control of Logistics Queueing Networks for Large-Scale Fleet Management', *Transportation Science*, vol.32, no.2, pp. 90-109.

Powell, W.B. & Carvalho, T.A. 1998b, 'Real-Time Optimization of Containers and Flatcars for Intermodal Operations', *Transportation Science*, vol.32, no.2, pp. 110-126.

Powell, W.B., Snow, W. & Cheung R. K. 2000, 'Adaptive Labeling Algorithm for the Dynamic Assignment Problem', *Transportation Science*, vol.34, no.1, pp. 50-66.

Powell, W.B., Towns, M.T. & Marar A. 2000, 'On the Value of Optimal Myopic Solution for Dynamic Routing and Scheduling Problems in the Presence of User Noncompliance', *Transportation Science*, vol.34, no.1, pp. 67-85.

Psarafitis, H.N. 1988, 'Dynamic Vehicle Routing Problems', in *Vehicle Routing: Methods and Studies,* eds. B.L. Golden & A.A. Assad, Elsevier Science Publishers, Amsterdam

Regan, A.C., Mahmassani, H.S. & Jaillet, P. 1995, 'Improving the Efficiency of Commercial Vehicle Operations Using Real-time Information: Potential Uses and Assignment Strategies', *Transportation Research Record*, vol.1493, pp. 188-198.

Regan, A.C., Mahmassani, H.S. & Jaillet, P. 1996a, 'Dynamic Decision Making for Commercial Fleet Operations Using Real-time Information', *Transportation Research Record*, vol.1537, pp. 91-97.

Regan, A.C., Mahmassani, H.S. & Jaillet, P. 1996b, 'Dynamic Dispatching Strategies Under Real-time Information for Carrier Fleet Management', in *Transportation and Traffic Theory,* eds. J.B. Lesort, pp.737-756.

Regan, A.C. 1997, Real-Time Information for Improved Efficiency Commercial Vehicle Operations. Ph.D. Thesis, The University of Texas at Austin.

Regan, A.C., Mahmassani, H.S. & Jaillet, P. 1998, 'Evaluation of Dynamic Fleet Management Systems: A Simulation Framework', *Transportation Research Record*, vol.1645, pp. 176-184.

Regan, A.C. & Golob, T.F. 1999, 'Freight Operators' Perceptions of Congestion Problems and the Application of Advanced Technologies: Results from a 1998 Survey of 1200 Companies Operating in California', *Transportation Journal*, vol.38, no.3, pp. 57-67.

Solomon M.M. 1987, 'Algorithm for the Vehicle Routing and Scheduling Problems with Time Window Constrains', *Operations Research*, vol.35, no.2, pp. 254-265.

Solomon, M.M. & Desrosiers, J. 1988, 'Time Window Constrained Routing and Scheduling Problems', *Transportation Science*, vol.22, no.1, pp. 1-13.

Wang, X. & Regan, A.C. 2001, 'Assignment Models for Local Truckload Trucking Problems with Stochastic Service Times and Time Window Constraints'*, TRB 80th Annual Meeting,* Washington D.C

Wolsey, L.A. 1998, *Integer Programming*, Hohn Wiley & Sons,

Yang, J., Jaillet, P. & Mahmassani, H.S. 1999, 'On-Line Algorithms for Truck Fleet Assignment and Scheduling under Real-Time Information', *Transportation Research Record*, vol.1967, pp. 107-113.

Yang, J., Jaillet, P. & Mahmassani, H.S. 2002, 'Study of a Real Time Multi-vehicle Truckload Pickup-and-Delivery Problem',

Yang, W.-H., Mathur, K. & Ballou, R.H. 2000, 'Stochastic Vehicle Routing Problem with Restocking', *Transportation Science*, vol.34, no.1, pp. 99-112.

You, P.-S. 1999, 'Dynamic Pricing in Airline Seat Management for Flights with Multiple Flight Legs', *Transportation Science*, vol.33, no.2, pp. 192-206.

United State 1997 Economic Census Transportation 1997 commodity flow survey, 1997 [Online], Available: http://www.bts.gov/ntda/cfs/97tcf-us.pdf [2003].

Comparison of Positions With and Without Selective Availability Full 24 Hour Data Sets, 2000 [Online], Available: http://www.ngs.noaa.gov/FGCS/info/sans_SA/compare/ERLA.htm [2002].

Dictionary of Algorithms, Data Structures, and Problems, [Online], Available: http://hissa.nist.gov/dads/ [2001].

Industry Report: Transportation Cargo, [Online], Available: http://www.activemedia -guide.com/print_transport.htm [2002].

Introduction to ITS/CVO Participant Manual Course 1, 1999 [Online], Available: http://www.itsdocs.fhwa.dot.gov/ [2002].

The Role of Transportation in Logistics, [Online], Available: http://wwwcf.fhwa.dot.gov/freightplanning/weart.htm [2002].

Statement By The President Regarding The United States' Decision To Stop Degrading Global Positioning System Accuracy, 2000 [Online], Available: http://www.ngs.noaa.gov/FGCS/info/sans_SA/docs/ statement.html [2002].

Truck Movements in America: Shipments from, to, within, and Through States, 1997 [Online], Available: http://www.bts.gov/transtu/ts1/ts1.htm [2001].

United State Department of Commerce News, 2003 [Online], Available: http://www.census.gov/mtis/www/current.html [2003].

Vita

Yong Jin Kim, the eldest son of Won Keun Kim and Kyu Won Cho was born in Seoul, Korea on December 11, 1969. A graduate of Jung-Dong High school in Seoul, Korea he received a Bachelor of Science from Seoul National University, Seoul, Korea in 1993, and a degree of Master of Science from the same university in 1996. From February 1994 to August 1995 he performed military duties as a private. Before pursuing his doctoral degree, he was employed as a part time lecturer at Dae-Jin University, Korea in Fall 1996 and Spring 1997. In August 1997 he entered the Graduate School of The University of Texas at Austin majoring in Transportation Systems. As a doctoral student at The University of Texas he has published several papers pertaining to his research and has made several presentations. He is the husband of Ji Yeon Kim since June 1998 and they have been blessed with baby boy Daniel Hojoong Kim in October 2001.

Permanent address: 4589 Via Marisol #261, Los Angeles, CA 90042

This dissertation was typed by the author