UNIVERSITÀ DELLA CALABRIA

Dottorato di Ricerca
in
"Ricerca Operativa" (MAT/09)
XX ciclo

Tesi di Dottorato

# Topics in real-time fleet management

**Emanuele Manni**

Supervisori
prof. Gianpaolo Ghiani

prof. Barrett W. Thomas

Coordinatore del corso di dottorato
prof. Lucio Grandinetti

Novembre 2007

UNIVERSITY OF CALABRIA

PhD
in
"Operations Research" (MAT/09)
cycle XX

PhD Thesis

# Topics in real-time fleet management

**Emanuele Manni**

Advisors
prof. Gianpaolo Ghiani

prof. Barrett W. Thomas

PhD Coordinator
prof. Lucio Grandinetti

November 2007

*Alla mia famiglia*
*e ad Alessandra.*

# Summary

The field of real-time fleet management has considerably grown during the last years, mainly due to recent developments in the economic and technologic sectors, allowing companies to focus on aspects not considered before, in order to be competitive on the market. Given these aspects, developing systems whereby goods are delivered to customers on time becomes crucial. Under this setting the key aspect is the availability of real-time information, like, for instance, vehicles location, road congestion, current fleet status, etc. For this purpose, advances in information technology and telecommunications, together with continually growing amount of data, offer opportunities for transportation companies to obtain this information with a relatively low effort. One of the most important tools available to companies is the ability to equip drivers with modern tools such as Global Positioning System (GPS) devices, eventually embedded into palmtop computers, in order to exactly know where the vehicles are and also to provide the drivers real-time directions.

In this thesis various issues concerning *real-time fleet management* are studied. The conventional vehicle routing problem (VRP) consists of determining a set of vehicle routes so that each customer of a set is visited exactly once and the total cost is minimized. Although dynamic VRPs represent important applications in many distribution systems, past research has focused mainly on static aspects of the VRPs. However, in the recent years, because of the rapid development in the telecommunications and computer hardware sectors, together with an increased focus on just-in-time logistics, a lot of efforts are being devoted to research on the more complex dynamic version of the VRP. The thesis begins by introducing the static VRP along with its dynamic counterpart (DVRP) and gives a discussion of the differences between the conventional static VRP and the dynamic VRP. Next, the main features of real-time vehicle routing problems are illustrated and a survey of the existing literature dealing with the dynamic vehicle routing problem as well as with a priori route design is presented.

The Dynamic and Stochastic Traveling Salesman Problem is introduced and an optimal policy through a Markov decision process is proposed as well as lower and upper bounds on the optimal policy cost are developed.

Then, an interesting question to answer in dynamic routing is whether or not it is worthwhile using complex procedures in order to design a priori routes. For this purpose, we consider several approaches for determining the a priori route for our dynamic routing problem and we determine under which circumstances the use of a priori routing can offer some value.

Next, we present the Dynamic and Stochastic Vehicle Dispatching Problem with Pickups and Deliveries, a real-time problem faced by local area (e.g., intra-city) courier companies serving same-day pickup and delivery requests for the transport of letters and small parcels. We develop anticipatory algorithms that evaluate alternative solutions through a short-term demand sampling and a fully sequential procedure for indifference zone selection.

Moreover, we introduce the same-day Courier Shift Scheduling Problem, a tactical problem faced by same-day courier companies, which amounts to minimize the staffing cost subject to probabilistic service level requirements.

Finally, we give our conclusions, as well as provide some directions for further research on real-time VRPs.

# Riassunto (In Italian)

Il settore della gestione di flotte in tempo reale ha visto un crescente aumento di interesse nel corso degli ultimi anni, principalmente grazie ai recenti sviluppi di natura tecnologica ed economica, che hanno consentito alle aziende del settore di spostare il proprio focus su aspetti fino a quel momento trascurati, con l'obiettivo di acquisire una maggiore competitività sul mercato. A questo proposito, un aspetto di cruciale importanza è la progettazione e lo sviluppo di sistemi di gestione di flotte in tempo reale in cui i beni sono consegnati ai clienti secondo modalità e tempi da loro richiesti. In questo contesto, un ruolo chiave è rappresentato dalla disponibilit in tempo reale di informazioni, quali, ad esempio, la posizione dei veicoli, la situazione di congestione stradale o lo stato corrente della flotta di veicoli. In questo senso, i grandi progressi effettuati nella tecnologia dell'informazione e delle telecomunicazioni, insieme alla sempre maggiore disponibilità di dati storici, offrono la possibilità alle aziende di trasporti di ottenere le informazioni volute con uno sforzo relativamente basso. Uno degli strumenti più importanti a disposizione delle aziende  la possibilità di fornire ai dipendenti dispositivi moderni quali, ad esempio, i GPS, eventualmente integrati in dispositivi palmari, con l'obiettivo di conoscere in ogni istante la posizione dei veicoli e di fornire indicazioni in tempo reale.

Obiettivo della presente tesi è analizzare diversi aspetti legati alla gestione di flotte in tempo reale. Il classico problema di instradamento dei veicoli (VRP) consiste nel determinare un insieme di rotte in modo tale che il costo complessivo sia minimizzato e che ciascun cliente sia visitato esattamente una volta. Benché il VRP abbia applicazioni importanti in una grande varietà di contesti di logistica distributiva, buona parte degli sforzi di ricerca degli ultimi decenni è stata indirizzata allo studio della versione statica del VRP. Occorre dire, comunque, che, nel corso degli ultimi anni, il crescente interesse nella logistica just-in-time e il rapido sviluppo tecnologico, hanno contribuito ad incrementare l'interesse nei confronti della versione dinamica (peraltro molto più complessa) del VRP. Nel presente lavoro di tesi si descrive il VRP statico, insieme alla sua controparte dinamica (DVRP), e si analizzano le differenze principali tra queste due varianti del problema. In seguito viene presentata una rassegna della letteratura esistente relativa a questa classe di problemi, con una particolare enfasi sul problema della determinazione di rotte a priori e sull'ottimizzazione in tempo reale.

Successivamente si introduce il problema del commesso viaggiatore dinamico e stocastico (Dynamic and Stochastic Traveling Salesman Problem) e, per mezzo di un processo decisionale markoviano, si individua una politica ottimale, oltre a fornire dei limiti inferiore e superiore al costo di tale politica.

Nel seguito si prova a rispondere ad un quesito molto attuale nei problemi dinamici di instradamento dei veicoli, relativo alla possibilità di utilizzare procedure più o meno complesse per la determinazione di rotte a priori per i veicoli. A tale scopo, vengono considerate diverse strategie a priori e si individuano le strategie che offrono le migliori prestazioni con riferimento al particolare problema di routing analizzato.

Nella sezione successiva si introduce il problema di assegnamento di richieste ai veicoli dinamico e stocastico con prelievo e consegna (Dynamic and Stochastic Vehicle Dispatching Problem with Pickups and Deliveries). Tale problema è affrontato in tempo reale dalle aziende di corrieri urbani che forniscono un servizio di prelievo e consegna di lettere o piccoli imballaggi nel corso della stessa giornata. Relativamente a tale problema, sono proposti degli algoritmi anticipativi che valutano diverse soluzioni alternative tramite un campionamento a breve termine della domanda futura ed una procedura completamente sequenziale per la selezione della zona di indifferenza.

In aggiunta, si introduce il problema di schedulazione dei turni per un'azienda di corrieri urbani (Same-day Courier Shift Scheduling Problem), un problema di natura tattica per questo tipo di aziende, che consiste nel minimizzare i costi del personale, dovendo rispettare dei vincoli di natura probabilistica sulla qualità del servizio.

Infine, si riportano le conclusioni e si individuano le possibile direzioni di ricerca futura.

# Acknowledgments

First of all, I am deeply indebted to my advisors, Prof. Gianpaolo Ghiani and Prof. Barrett Thomas. To each of them I owe a great debt of gratitude for their patience, inspiration and friendship. Thank you Gianpaolo for proposing me a promising research subject and motivating me to work on challenging problems. You came down to my level in order to collaborate with me, you was a real partner, before being a master. Thank you Barry for all the guiding, cooperation, encouragements, lasting support, and for making me feel like at home when I visited you at the University of Iowa.

Next, I would like to thank all the colleagues at the Logistics Lab, especially Adriana, Marco and Matteo, for being real friends more than colleagues. We spent many enjoyable hours together.

Thanks also to my family who have been extremely understanding and supporting. They sustained me both in my studies and life with their love and their presence and put an immense trust in me. Thanks for everything. A particular thanks goes to my brother Andrea for sharing with me these last three years as PhD students.

Finally, last but certainly not least I wish to thank Alessandra for all her love, support and comprehension throughout all these years. Thanks for encouraging me with the right words when I needed them, thanks for being there when I needed you.

Emanuele Manni
November 2007

# Contents

# Chapter 1

# Introduction

*Gutta cavat lapidem; non vi, sed saepe cadendo.*
*Huic addiscit homo: non vi, sed saepe legendo.*
*Anonymous*

*Vehicle Routing Problems* (VRPs) are central to logistics management both in the private and public sectors. They consist of determining optimal vehicle routes through a set of users, subject to side constraints. The most common operational constraints impose that the total demand carried by a vehicle at any time does not exceed a given capacity, the total duration of any route is not greater than a prescribed bound, and service time windows set by customers are respected. In long-haul routing, vehicles are typically assigned one task at a time while in short-haul routing, tasks are of short duration (much shorter than a work shift) and a tour is to be built through a sequence of tasks. For a survey on the most relevant modeling and algorithmic issues on VRPs, see the recent book by Toth and Vigo (2002).

There exist several important problems that must be solved in real time. In what follows, we review the main applications that motivate the research in the field of the real-time VRPs.

**Dynamic fleet management.** Several large scale trucking operations require real-time dispatching of vehicles for the purpose of collecting or delivering shipments. Important savings can be achieved by optimizing these operations (Brown and Graves, 1981, Brown et al., 1987, Powell, 1986, 1990, Goetschalckx, 1988, Rego and Roucairol, 1995, Savelsbergh and Sol, 1998).

**Vendor-managed distribution systems.** In vendor-managed systems, distribution companies estimate customer inventory level as to replenish the customers before they run out of stock. Hence, demands are known beforehand in principle and all customers are *static*. However, because demand has always a random component, some customers (usually a small percentage) may run out of stock and have to be serviced urgently (Campbell et al., 1998, Larsen, 2000).

**Couriers.** Long-distance couriers need to collect locally outbound parcels before sending them to a remote terminal to consolidate loads. Also, loads coming from remote terminals have to be distributed locally. Most pick-up requests are dynamic and have to be serviced the same day if possible (Gendreau et al., 1999, 2006, Cordeau et al., 2002).

**Rescue and repair service companies.** There are several companies providing rescue or repair services (broken car rescue, appliance repair, etc.) (Madsen et al., 1995b, Weintraub et al., 1999, Brotcorne et al., 2003).

**Dial-A-Ride Systems.** Dial-a-ride systems provide transportation services to people between given origin-destination pairs. Customers can book a trip one day in advance (*static customers*) (Cordeau and Laporte, 2003) or make a request at short notice (*dynamic customers*) (Roy et al., 1984, Madsen et al., 1995a).

**Emergency Services.** Emergency services comprise police, fire fighting and ambulance services. By definition, all customers are dynamic. Moreover, the demand rate is usually low so that vehicles become idle from time to time. In this context, relocating idle vehicles in order to anticipate future demands or to escape from downtown rush hour traffic jam is an important consideration (Psaraftis, 1980, Gendreau et al., 1997, 2001).

**Taxi Cab Services.** In taxi cab services, almost every customer is dynamic. As in emergency services, relocating temporary idle vehicles is a consideration.

Due to recent advances in information and communication technologies, vehicle fleets can now be managed in real-time. When jointly used, devices like Geographic Information Systems (GIS), Global Positioning Systems (GPS), traffic flow sensors, and cellular telephones are able to provide relevant real-time data, such as current vehicle locations, new customer requests and periodic estimates of road travel times (Brotcorne et al., 2003). If suitably processed, this large amount of data can be used to reduce cost and improve service level. To this end, revised routes have to be generated as soon as new events occur.

In recent years, three main developments have contributed to the acceleration and quality of algorithms relevant in a real-time context. The first is the increase in computing power due to better hardware. The second is the development of powerful metaheuristics whose main impact has been mostly on solution accuracy even if this gain has sometimes been achieved at the expense of computing time. The third development has arisen in the field of parallel computing. The combination of these three features has yielded a new generation of powerful algorithms that can effectively be used to provide real-time solutions in dynamic contexts.

## 1.1 Outline of the thesis

This thesis discusses various issues concerning *real-time fleet management.* In the present chapter, we introduce the vehicle routing problem (VRP) along with its dynamic counterpart (DVRP), and we give a discussion of the differences between the conventional static VRP and the dynamic VRP. Next, we illustrate the main features of real-time vehicle routing problems. Moreover, we present the notion of degree of dynamism, first introduced by Lund et al. (1996) and then extended by Larsen (2000). We list the different objectives that are to be achieved in dynamic problems. In addition, we present the different hypothesis on the spacial distribution of demands together with different possibilities for comparing routing and dispatching policies. The chapter closes with a survey of the existing literature dealing with dynamic routing problems. The emphasis of the section is on both a priori and real-time optimization methods.

In chapter 2, we introduce the Dynamic and Stochastic Traveling Salesman Problem and propose an optimal policy through a Markov Decision Process as well as develop lower and upper bounds on the optimal policy cost.

In chapter 3, we present several strategies for implementing a priori routes within a dynamic routing context, and we identify situations in which the use of more involved a priori strategies can give some benefit.

In chapter 4, we consider the Dynamic and Stochastic Vehicle Dispatching Problem with Pickups and Deliveries, a real-time problem faced by local area (e.g., intra-city) courier companies serving same-day pickup and delivery requests for the transport of letters and small parcels. We develop anticipatory algorithms that evaluate alternative solutions through a short-term demand sampling and a fully sequential procedure for indifference zone selection. A peculiar feature of this procedures is that they allow the elimination, at an early stage of experimentation, those solutions that are clearly inferior, thus reducing the overall effort to select the best.

In chapter 5, we consider the same-day Courier Shift Scheduling Problem, a tactical problem faced by same-day courier companies, which amounts to minimizing the staffing cost subject to probabilistic service level requirements.

Then, in chapter 6, we give our conclusions in a brief summary of the discussions of this thesis as well as provide some directions for further research on real-time VRPs.

## 1.2 Static vs Dynamic Vehicle Routing Problems

A vehicle routing problem is said to be *static* if its input data (for example, travel times and demands) do not depend explicitly on time, otherwise it is *dynamic*. Moreover, a VRP is *deterministic* if all input data are known when designing vehicle routes, otherwise it is *stochastic*.

**Static VRPs.** A static problem can be either deterministic or stochastic. In deterministic and static VRPs (like the classical Capacitated VRP surveyed in Toth and Vigo (2002)) all data are known in advance and time is not taken into account explicitly. In stochastic and static VRPs (Laporte and Louveaux, 1998), vehicle routes are designed at the beginning of the planning horizon, *before* uncertain data become known. Uncertainty may affect which service requests are present, user demands, user service times or travel times. If input data are uncertain, it is usually impossible to satisfy the constraints for all realizations of the random variables. If uncertainty affects the constraints but the objective function is deterministic, it can be required that constraints be satisfied with a given probability (*Chance Constrained Programming*, CCP). In a more general approach, a first phase solution is constructed *before* uncertain data are available and corrective (or *recourse*) actions are taken at a second stage once all the realisations of the random variables become known. The objective to be minimized is the first stage cost plus the expected recourse cost (*Stochastic Programming with Recourse*, SPR).

**Dynamic VRPs.** A dynamic problem can also be deterministic or stochastic (Powell et al., 1995). In deterministic and dynamic problems, all data are known in advance and some elements of information depend on time. For instance, the VRP with time windows reviewed in Cordeau et al. (2001) belongs to this class of problems. Similarly, the *Traveling Salesman Problem* (TSP) with time-dependent travel times (Malandraki and Daskin, 1992) is deterministic and dynamic. In this problem, a traveling salesperson has to find the shortest closed tour among several cities passing through all cities exactly once, and travel times may vary throughout the day. Finally, in stochastic and dynamic problems (also known as *real-time* routing and dispatching problems) uncertain data are represented by *stochastic processes.* For instance, user requests can behave as a Poisson process (as in Bertsimas and Van Ryzin (1991)). Since uncertain data are gradually revealed during the operational interval, routes are *not* constructed beforehand. Instead, user requests are dispatched to vehicles in an on-going fashion as new data arrive (Psaraftis, 1988). The *events* that lead to a plan modification can be: (i) the arrival of new user requests, (ii) the arrival of a vehicle at a destination, (iii) the update of travel times. Every event must be processed according to the policies set by the vehicle fleet operator. As a rule, when a new request is received, one must decide whether it can be serviced on the same day, or whether it must be delayed or rejected. If the request is accepted, it is temporarily assigned to a position in a vehicle route. The request is effectively serviced as planned if no other event occurs in the meantime. Otherwise, it can be assigned to a different position in the same vehicle route, or even dispatched to a different vehicle. It is worth noting that at any time each driver just needs to know his next stop. Hence, when a vehicle reaches a destination it has to be assigned a new destination. Because of the difficulty of estimating the current position of a moving vehicle, reassignments could not easily made until quite recently. Due to advances in communication technologies, however, route diversions and reassignments are now a feasible option and can result in a cost saving or in an improved service level (Gendreau and Potvin, 1998, Ichoua et al., 2000). Finally, if an improved estimation of vehicle travel times is available, it may be useful to modify the current routes or even the decision of accepting a request or not. For example, if an unexpected traffic jam occurs, some user services can be deferred. It is worth noting that when the demand rate is low, it is useful to relocate idle vehicles in order to anticipate future demands or to escape a forecasted traffic congestion.

## 1.3 Characterization of Dynamic Vehicle Routing Problems

As pointed out by Psaraftis (1988, 1995) dynamic VRPs possess a number of peculiar features, some of which have just been described. In this section, we more fully characterize the dynamic VRP.

### 1.3.1 The degree of dynamism of a problem

Designing a real-time routing algorithm depends to a large extent on how dynamic the problem is. To quantify this concept, Lund et al. (1996) and Larsen (2000) have defined the *degree of dynamism* of a problem.

Without loss of generality, we assume that the planning horizon is a given interval $[0,T]$, possibly divided into a finite number of smaller intervals. Let $n_s$ and $n_d$ be the number of static and dynamic requests, respectively. Moreover, let $t_i \in [0,T]$ be the *occurrence time* of service request $i$. Static requests are such that $t_i = 0$ while dynamic ones have $t_i \in (0,T]$. Lund et al. (1996) define the degree of dynamism $\delta$ as:

$$\delta = \frac{n_d}{n_s + n_d},$$

which may vary between 0 and 1. Its meaning is straightforward. For instance, if $\delta$ is equal to 0.3, then 3 customers out of 10 are dynamic. In his doctoral thesis, Larsen (2000) generalizes the definition proposed by Lund et al. (1996) in order to take into account both dynamic request occurrence times and possible time windows. He observes that, for a given $\delta$ value, a problem is more dynamic if immediate requests occur at the end of the operational interval $[0, T]$. As a result he introduces a new measure of dynamism:

$$\delta' = \frac{\sum\limits_{i=1}^{n_s+n_d} t_i/T}{n_s + n_d}.$$

It is worth noting that $\delta'$ ranges between 0 and 1. It is equal to 0 if all user requests are known in advance while it is equal to 1 if all user requests occur at time $T$. Finally, Larsen (2000) extends the definition of $\delta'$ to take into account possible time windows on user service time. Let $a_i$ and $b_i$ be the *ready time* and *deadline* of client

$i$ ($t_i \leq a_i \leq b_i$), respectively. Then,

$$\delta'' = \frac{\sum_{i=1}^{n_s+n_d} [T - (b_i - t_i)]/T}{n_s + n_d}.$$

It can be shown that $\delta''$ also varies between 0 and 1. Moreover, if no time windows are imposed (i.e., $a_i = t_i$ and $b_i = T$), then $\delta'' = \delta'$. As a rule, vendor-based distribution systems (such as those distributing heating oil) are weakly dynamic. Problems faced by long-distance couriers and appliance repair service companies are moderately dynamic (Larsen et al., 2002). Finally, emergency services, taxi cab services or same-day urban couriers exhibit a strong dynamic behavior.

### 1.3.2 Objectives

In real-time routing problems the objective to be optimized is often a combination of different measures. Larsen (2000) observes that in weakly dynamic systems the focus is on minimizing routing cost. On the other hand, when operating a strongly dynamic system, minimizing the expected *response time* (i.e., the expected time lag between the instant a user service begins and its occurrence time) becomes a key issue. Another meaningful criterion which is often considered (alone or combined with other measures) is throughput optimization, the maximization of the expected number of requests serviced within a given period of time (Psaraftis, 1988). A completely different approach is followed in Potvin et al. (1992, 1993). In particular, the former paper is based on the assumption that in most real-time routing problems the objective is fuzzy. Hence, a suitably trained neural network is used to reproduce automatically the dispatching decisions of skilled personnel.

### 1.3.3 Spatial distribution and time pattern of user requests

As explained later in this section, better real-time dispatching and routing decisions can be made if uncertain data estimations (derived from historical data) are used. In several papers (see, Bertsimas and Van Ryzin, 1991, 1993, Swihart and Papastavrou, 1999, Papastvrou, 1996), it is supposed that user requests are uniformly distributed in a convex bounded Euclidean region, and occur according to a Poisson process with arrival rate $\lambda$. In problems with pick-ups and deliveries (see, e.g., Swihart and

Papastavrou (1999)), it is also assumed that the delivery locations are independent of the pick-up locations.

### 1.3.4 Comparing different dispatching and routing procedures

Evaluating a dispatching and routing algorithm can be done analytically if specific assumptions are satisfied (see, Bertsimas and Van Ryzin (1991) where demands are modeled as a Poisson process). If these hypotheses do not hold, algorithmic performances have to be evaluated empirically through discrete-time simulation, as in Gendreau et al. (1999). Another way to assess the performance of a heuristic for a dynamic routing problem is to run it independently on the corresponding static data assuming all information is available when planning takes place. By comparing the static and dynamic solution one can compute the *value of information* as is typically done in decision trees. Such an approach was used by Mitrović-Minić et al. (2004).

## 1.4 Literature review

The aim of this section is to survey the existing literature on dynamic vehicle routing problem and related problems. In recent years, we observe a growing number of papers treating stochastic and/or dynamic vehicle routing problems. The literature on dynamic vehicle routing problems covers many different applications and methodological approaches. Naturally, this section cannot cover all aspects of vehicle routing problems with dynamic or stochastic elements. The goal of this section is rather to provide a brief survey of the literature related to these problems, with a particular emphasis for the topics covered in this thesis. More specifically, we focus on a priori and real-time optimization methods. The study of the literature ended October 31st 2007 and work published after this date is not treated in this thesis. This section is organized as follows. In subsection 1.4.1, we discuss the literature related to a priori routing problems, whereas in subsection 1.4.2, we turn to problems using real-time optimization.

### 1.4.1 A Priori Optimization Based Methods

Perhaps the best known a priori routing problem is the probabilistic traveling salesman problem (PTSP). In the PTSP, a set of customers with known probabilities of needing service on a given day are routed with the objective of minimizing the expected travel distance. On a given day, customers who do not require service are skipped in the tour. Thus, while customer demand is essentially revealed dynamically over time, the problem is a static one. The problem was formally introduced by Jaillet (1988) with Bertsimas (1988) adding further foundational work. Because exact approaches have a limited ability to solve PTSP problems (Laporte et al., 1994), much of the work in this area focuses on heuristic solution approaches. An overview of the current state of PTSP research can be found in Campbell and Thomas (2007 forthcoming). Campbell and Thomas (2007a,b) discuss an extension of the PTSP which includes delivery deadlines.

Early work on dynamic and stochastic routing routing problems focuses on the dynamic routing and dispatching problem (DRDP). In DRDPs, stochastic information about future requests is typically ignored and the dynamic nature of the problem is not acknowledged in the solution approach. Rather, the research presented in these papers can be considered reactive in that requests for service are considered only when they occur and no effort is made to anticipate that future requests will occur. In this thesis, the a priori routing strategy discussed in Section 3.4.1 is similar to those found in the DRDP in that it ignores any information about future requests. However, our handling of dynamic requests over the course of the problem does anticipate the late-request customers. Overviews of the DRDP and related literature can be found in Powell et al. (1995) and Psaraftis (1988, 1995). A more modern approach to the DRDP can be found in Gendreau et al. (1999) and Ichoua et al. (2000) in which a parallel implementation of a tabu search is used to continuously update vehicle routes as new requests occur.

More recent work attempts to account for potential future requests through the use of dynamic heuristic strategies coupled with a priori tours. Often, the approaches form a priori tours of any static customers and then insert dynamic requests as they occur. In contrast to the DRDP, strategies, notably waiting in various locations on the tours, are used to account for the likelihood of future requests. In the current literature, however, most a priori tours are constructed without regard to the locations or likely locations of future requests. Instead, the static customers are typically routed via some non-dynamic procedure for the vehicle routing problem. Kilby et al. (1998) use such an a priori procedure to produce routes which are then coupled with a sampling procedure to create an dynamic routing heuristic. Larsen et al. (2002, 2004) consider a problem in which some service requests are known in advance of the start of service. They route these advance-requests using algorithms for the traveling salesman problem with time windows, again ignoring the possibility of future requests in the creation of these routes. The authors then couple these routes with various heuristics that account for dynamic requests. Likewise, Branke et al. (2005) routes advance-request customers using methodologies for the vehicle routing problem in order to construct a priori routes which are then used in conjunction with waiting strategies and insertion techniques.

As we do in Chapter 3, Thomas (2007) consider the case where the locations of both static and dynamic customers are known in advance. The author builds the a priori route for the static customers by constructing the shortest path through them. Moreover, Thomas extrapolates the structure for the optimal policy for one dynamic customer to develop a real-time heuristic that performs well when the percentage of dynamic customer is 25% or less. The author shows that a strategy that distributes waiting time across static customer locations works well as the percentage of dynamic customers increases. In chapter 3, we use the second dynamic strategy in order to anticipate then serve dynamic customers.

The use of information about future requests in the construction of a priori routes for dynamic problems can be found in Secomandi (2000) and Secomandi (2001). In these works, an priori route is constructed for the traditional stochastic vehicle routing problem. This a priori route is then used as an initial heuristic policy for neuro-dynamic programming algorithms. The algorithms are applied to a version of the stochastic vehicle routing problem with stochastic demand for which the original a priori route is dynamically updated based on realized demand. In Section 3.4.4, we also consider the construction of the a priori routes using probabilistic information about future requests.

Particularly relevant to the work in this thesis are the articles by Mitrović-Minić et al. (2004) and Mitrović-Minić and Laporte (2004). Mitrović-Minić and Laporte (2004) consider a dynamic pick-up and delivery problem. Analogous to the discussed vehicle routing problems, they construct initial a priori tours into which dynamically occurring customers are inserted. They examine four waiting strategies for the dynamic *Pickup and Delivery Problem with Time Windows* (PDPTW). In the dynamic PDPTW, the presence of time windows allows the vehicles to wait at various locations along their routes. The authors show that an adequate distribution of the waiting time may affect the planner's ability to make good decisions at a later stage. Mitrović-Minić et al. (2004) propose double-horizon based heuristics. These procedures make use of two different planning horizons (short term and long term) to which different goals are applied. The idea is that in the short term, it is important to concentrate on minimizing total route length, whereas in the long term, it is more crucial to preserve a certain amount of flexibility in order to better accommodate future requests.

### 1.4.2 Real-time Optimization Methods

In this section, we review literature for real-time fleet management problems, a broad category in which vehicle routes are built in an on-going fashion as customer requests, vehicle locations and travel times are revealed over the planning horizon. Overviews of these problems can be found in Psaraftis (1995), Gendreau and Potvin (1998), Ghiani et al. (2003), and Larsen et al. (2007, forthcoming).

Reactive procedures are, for instance, those described by Gendreau et al. (2006). The authors propose adaptive descent and tabu search heuristics that utilize a neighborhood structure based on ejection chains. Numerical results show that when enough computing power is available, they produce improved results over simpler heuristics (even if the optimization takes place over known requests only, with no consideration for the future).

Assuming that some probabilistic knowledge is available, two different ways of exploiting this information are reported in the literature: analytical studies and anticipatory algorithms. The former approach examines dispatching and routing policies whose performance can be determined analytically if specific assumptions are satisfied. Along this line of research, Bertsimas and Van Ryzin (1991, 1993) identify optimal policies both in light and heavy traffic, whereas demands are distributed in a bounded area in the plane and arrivals are modelled as a Poisson process. Papastvrou (1996) describes a routing policy that performs well both in light and heavy traffic, while Swihart and Papastavrou (1999) examine a dynamic pickup and delivery extension. We refer the reader to Larsen et al. (2007, forthcoming) for an in-depth description of these articles.

The latter approach aims at developing heuristic anticipatory algorithms that incorporate explicitly currently available information about future events. A seminal contribution in this research area is Powell et al. (1988), whose work was motivated by long-haul truckload trucking applications. These results have been subsequently extended in Godfrey and Powell (2002), Powell and Topaloglu (2003) and Spivey and Powell (2004). Other related contributions are Larsen (2000) and Larsen et al. (2002), Bent and Van Hentenryck (2004), van Hemert and La Poutré (2004), Thomas and White III (2004), Ichoua et al. (2006), Hvattum et al. (2006), and Ghiani et al. (2007). Larsen (2000) and Larsen et al. (2002) describe some policies for relocating idle vehicles in anticipation of future demands. Bent and Van Hentenryck (2004) consider a vehicle routing problem with hard time windows where customer locations and service times are random variables which are realized dynamically during plan execution. They develop a multiple scenario approach which continuously generates plans consistent with past decisions and anticipating future requests. The current solution (or distinguished plan) is selected by a consensus function (Stefik, 1981) that chooses the solution most similar to a continuously updated pool of routings. No detail has been provided on the number of samples to be taken for each alternative solution. To exploit probabilistic information about future service requests, van Hemert and La Poutré (2004) introduce the concept of fruitful regions in a dynamic routing context. Fruitful regions are regions that have a high potential of generating loads to be transported. The authors define potential schedules by sampling fruitful regions and have provided an evolutionary algorithm for deciding whether to move or not towards one of such regions in anticipation of future demand. Waiting strategies have not been addressed explicitly. Thomas and White III (2004) introduce a problem in which the objective is to maximize rewards being received for serving customers minus costs incurred for traveling along arcs. They model the problem as a finite-horizon Markov decision process and perform numerical experiments in order to compare the optimal policy with a reactive strategy that ignores potential customer requests. Ichoua et al. (2006) develop a parallel tabu search heuristic that allows a vehicle to wait in its current zone if the probability of a future request reaches a particular threshold. Idle vehicle relocation is not dealt with. Hvattum et al. (2006) introduce a hedging heuristic that uses sampling and common features of deterministic routes constructed for the sampled customers to build a plan for each time interval in the time horizon. According to the authors, this approach requires some computation and is not necessarily implementable in a real-time setting which requires quick decisions. Ghiani et al. (2007) develop an anticipatory mechanism that evaluates alternative solutions through a short-term demand sampling and a fully sequential procedure for indifference zone selection. The authors address a number of issues involved in real-time fleet management, like assigning requests to vehicles, routing the vehicles, scheduling the routes and relocating idle vehicles. Computational results show that the anticipatory mechanism allows to yield consistently better solutions than a purely reactive procedure.

# Chapter 2

# The Dynamic and Stochastic Traveling Salesman Problem

## 2.1 Introduction

The purpose of this chapter is to introduce the *Dynamic and Stochastic Traveling Salesman Problem* (DSTSP) as well as to study exact and heuristic waiting policies for it. The DSTSP is defined on a graph $G = (V,A)$, where $V$ is a vertex set and $A$ is an arc set. A vehicle based at a depot $i_0$ has to service a number of pick-up requests, or delivery requests, but not both. Request $i_k \in V' \subseteq V$ $(k = 1,\dots,n)$ may arise at time instant $T_k$ with probability $p_k$. A customer $i_k$ may not require service if $p_k < 1$. Time instants $T_k$ $(k = 0,\dots,n)$, with $T_0 = 0$, are assumed to be integer and the requests are supposed to be statistically independent. The vehicle may wait at any vertex (both a customer or a non-customer vertex) in order to anticipate future demand. It is worth noting that, unlike what happens in the classical (static) *Traveling Salesman Problem*, in which the vehicle follows a shortest path between two consecutive customers, in the DSTSP the vehicle may wait for some time at some vertices outside the current route. This may be useful, for example, to promptly collect enough demand in an area before moving in another part of the service territory. Let $t_{ij}$ be the shortest travel time from vertex $i \in V$ to vertex $j \in V$. As is common in dynamic vehicle routing problems, the aim is to maximize overall customer service level rather than minimize the total traveled distance. Let $\tau_k$ be the service time of a customer $i_k$ requiring service. To each customer is associated a non-decreasing and convex function $f_k(\tau_k)$ expressing the penalty associated with customer $i_k$. This definition includes the case in which $f_k(\tau_k)$ represents customer *waiting time* (i.e., $f_k(\tau_k) = \tau_k - T_k, \tau_k \geq T_k$) or a more involved penalty function (e.g., $f_k(\tau_k) = 0$ if $T_k \leq \tau_k \leq D_k$ and $f_k(\tau_k) = \tau_k - D_k$ if $\tau_k \geq D_k$, where $D_k$ is a soft deadline associated with customer $i_k$). Decision epochs occur at time instants $T_k$ ($k = 1, \ldots, n$). The DSTSP consists of determining a policy such that at any epoch a decision is made: a) on the order in which pending customers have to be visited; b) on how to reposition the vehicle in anticipation of future demand. The latter issue includes deciding how long the vehicle has to wait at various locations along its route as well as whether to relocate the vehicle to some vertices outside the current route. The objective function to be minimized is the expected total penalty:

$$z = \sum_{k=1}^{n} E[f_k(\tau_k)|k]p_k \tag{2.1}$$

where $E[f_k(\tau_k)|k]$ is the expected penalty associated to customer $i_k$ requiring service. Assume that the order in which customers are serviced is given ($i_1 \leq i_2 \leq \ldots \leq i_n$ without any loss of generality) and we develop exact and heuristic waiting policies for the DSTSP.

## 2.2 A lower bound on the optimal policy expected penalty

In this section a lower bound on the expected penalty of the optimal policy is computed under the hypothesis of perfect information, i.e., when all occurring requests are known at the beginning of the planning horizon. Let $\sigma(r)$ be the $r$-th occurring request ($r = 1, \ldots, m \leq n$) and $\sigma(0) = 0$. Under perfect information an optimal policy can be devised straightforwardly. Indeed the vehicle should drive immediately to the first occurring customer, then if $t_{i_{\sigma(0)} i_{\sigma(1)}} < T_{\sigma(1)}$ wait until $T_{\sigma(1)}$, service $i_{\sigma(1)}$, then drive to $i_{sigma(2)}$, etc. Let $\tau'_k$ be the service time of customer $i_k$ under an optimal policy in case of perfect information. For every realization of the demand (such that request $i_k$ occurs), the following inequalities are valid:

$$\tau_k \geq \tau'_k \geq \sum_{j=0,\ldots,m-1:\ \sigma(j+1) \leq k} t_{i_{\sigma(j)} i_{\sigma(j+1)}} \tag{2.2}$$

This relationship holds since $\tau'_k$ is the right-hand side of (2.1), plus the waiting times at the occurring customers $i_{\sigma(j)}$ $(\sigma(j) \leq k)$. Consequently, assuming request $i_k$ is issued, the expected value of the right-hand side of (2.2) is a lower bound on the expected value $E[\tau_k]$. The probability associated with $t_{i_{\sigma(j)}i_{\sigma(j+1)}}$ in this expected value computation is the probability that both $t_{i_{\sigma(j)}}$ and $t_{i_{\sigma(j+1)}}$ occur and no intermediate customers issue an order:

$$p_{\sigma(j)\sigma(j+1)}(1 - p_{\sigma(j+1)})(1 - p_{\sigma(j+2)}) \ldots (1 - p_{\sigma(j+1)-1}), \qquad \sigma(j+1) < k$$
$$p_{\sigma(j)\sigma(j+1)}(1 - p_{\sigma(j+1)})(1 - p_{\sigma(j+2)}) \ldots (1 - p_{\sigma(j+1)-1}), \qquad \sigma(j+1) = k$$

where we have assumed $p_{\sigma(0)} = p_0 = 1$. Hence,

$$E[\tau_k] \geq \sum_{r=0}^{k-2} \sum_{s=r+1}^{k-1} \left[ t_{rs} p_r p_s \prod_{u=r+1}^{s-1} (1 - p_u) \right] + \sum_{r=0}^{k-1} \left[ t_{rk} p_r \prod_{u=r+1}^{k-1} (1 - p_u) \right] \tag{2.3}$$

Let $L_k$ be the right-hand side of 2.3. Based on the Jensen inequalities (Birge and Louveaux, 1997) and the monotonicity of penalty functions $f_k()$, we can write:

$$E[f_k(\tau_k)] \geq f_k(E[\tau_k]) \geq f_k(L_k)$$

We then obtain the required lower bound:

$$LB = \sum_{k=1}^{n} f_k(L_k) p_k \tag{2.4}$$

It is worth noting that this lower bound requires $O(n^3)$ computations provided that functions $f_k()$ $(k = 1, \ldots, n)$ can be evaluated in constant time.

## 2.3 Heuristic policies

In this section we assess the expected penalty of two heuristic policies, called *Wait-First* (WF) and *Drive-First* (DF), introduced by Mitrović-Minić and Laporte (2004) in a purely dynamic setting. The WF strategy requires an idle vehicle to wait at its current location until a new customer request arrives, while the DF strategy requires an idle vehicle to drive to its next potential customer. Under a WF policy, the time between the service of customers $i_r$ and $i_s$ $(s > r)$ is equal to

$$A_{rs}(\tau_r) = \max\{T_s - \tau_r, 0\} + t_{i_r i_s}$$

provided that customer $i_r$ is serviced at time $\tau_r$ and no intermediate request is issued. Indeed, $\max\{T_s - \tau_r,0\}$ represents the waiting time at vertex $i_r$ whereas $t_{i_r i_s}$ is the travel time between the two vertices. The probability that the service time $\tau_k$ is equal to $t$ under a WF policy can be computed through the iterative formula:

$$\Pr(\tau_k = t|k) = \sum_{l<k} \sum_{t'<t-t_{lk}} \left[ \Pr(\tau_l = t - A_{lk}(t')|l) \prod_{r=l+1}^{k-1} (1-p_r) \right] \tag{2.5}$$

with the initialization $Pr(\tau_0 = 0) = 1$. Once these probabilities have been computed we can calculate the expected value of the total penalty associated to the WF policy by applying the definition:

$$z_{WF} = \sum_{k=1}^{n} p_k \sum_t f_k(t) \Pr(\tau_k = t|k) \tag{2.6}$$

Under a DF policy, the previous procedure still applies, except that the computation of $A_{km}(\tau_k)$ is more elaborate. If the vehicle services customer $i_k$ at time $\tau_k$, then it moves along a shortest path from $i_k$ to $i_{k+1}$. Let $i'_r$ be the vertex where the vehicle is located at time instant $T_r$ $(r = 1,\ldots,n)$. If $\tau_k + t_{i_k i_{k+1}} \leq T_{k+1}$, then $i'_{k+1} = i_{k+1}$, where the vehicle waits for $\max\{T_{k+1} - \tau_k - t_{i_k i_{k+1}},0\}$ time instants. Otherwise, $i'_{k+1}$ is a vertex along a shortest path from $i_k$ to $i_{k+1}$, where the vehicle is diverted to $i_{k+2}$. In any case the vehicle then follows a shortest path from $i'_{k+1}$ to $i_{k+2}$. By iteratively applying these procedures, vertices $i'_{k+1},\ldots,i'_m$ are identified and $A_{rs}(\tau_r)$ is computed as

$$A_{rs}(\tau_r) = t_{i_r i' r+1} + \max\{T_{r+1} - \tau_r - t_{i_r i_{r+1}},0\} + \sum_{j=r+2}^{s} \left( t_{i'_j i'_{j+1}} + \max\{T_j - T_{j-1} - t_{i'_{j-1} i_j},0\} \right) \tag{2.7}$$

Formula (2.7) can be used to compute the expected penalty associated with the DF policy through relations (2.5) and (2.6).

## 2.4 A Markov Decision Process

We now determine the optimal policy through a Markov Decision Process (MDP) which is a well-known approach for modeling and solving dynamic and stochastic decision problems. Much has been written about dynamic programming. Some recent books in this area are Puterman (1994), Bertsekas (1995), and Sennot (1999).

In our MDP, decisions are made at time instants $T_k$ ($k = 1,\dots,n$) (*decision epochs*) at which it becomes known whether or not customers need service. In particular, at time $T_k$ we have to decide, in case the vehicle becomes idle before $T_{k+1}$, to which vertex the vehicle should be repositioned at time $T_{k+1}$.

A fundamental concept in MDPs is that of a *state*, denoted by $s$. The set $S$ of all possible states is called the *state space*. The decision problem is often described as a controlled stochastic process that occupies a state at each point in time. The state should be a sufficient and efficient summary of the available information affecting the future of the stochastic process. In our problem, at every time instant $T_k$ (*stage*) the state is represented by the triple $(T_k,i^*_k,t^*_k)$, where $i^*_k$ and $t^*_k$ are respectively the vertex and time where the vehicle will become idle (i.e., with no pending requests) at the next epoch. Let $S_k$ be the set of possible states at stage $T_k$. Obviously, $S = \bigcup_{k=0,\dots,n} S_k$. The set $S_0$ contains a single state $s = (T_0,i_0,T_0)$ since the vehicle is idle at the depot at time $T_0 = 0$. At any stage $T_k$, we first know whether customer $i_k$ requires service, and we may then decide how to reposition the vehicle. Let $s = (T_k,i'_k,t'_k) \in S_k$ be the state before information about $i_k$ becomes known (*chance state*). If this request occurs, $i_k$ is appended to the route so that the state becomes $(T_k,i''_k,t''_k)$ with $i''_k = i'_k$ and $t''_k = t'_k + t_{i'_k i_k}$. Otherwise, the state remains unchanged ($i''_k = i'_k$ and $t''_k = t'_k$). These two states $(T_k,i''_k,t''_k)$ are called the *decision states* associated with chance state $s$. Let $V^+(i,\Delta t)$ be the set of vertices that can be reached from $i \in V$ within no more than $\Delta t(> 0)$ time units, and let $V^+(i,\Delta t) = \{i\}$ if $\Delta t \leq 0$. Hence, once it is known whether request $i_k$ has occurred, we can reposition the vehicle to a vertex $i'_{k+1} \in V^+(i''_k,T_{k+1} - t''_k)$ where the vehicle will arrive at time $t'_{k+1} = t''_k + t_{i''_k i'_{k+1}}$. Consequently, at stage $T_{k+1}$, the state may be chosen from the subset $\{(T_{k+1},i'_{k+1},t'_{k+1}) : i'_{k+1} \in V^+(i''_k,T_{k+1} - t''_k),t'_{k+1} = t''_k + t_{i''_k i'_{k+1}}\} \subseteq S_{k+1}$. It is worth noting that, if the $k$-th request $i_k$ occurs, the service time $\tau_k$ of customer $i_k$ is then equal to $t''_k = t'_k + t_{i'_k i_k}$. Let $z_s$ be the expected penalty $p_k f_k(\tau_k)$ associated with the service of customer $i_k$ if the vehicle is in state $s \in S_k$ and let $Z_s$ be the total expected penalty associated to an optimal policy servicing customers $\{i_k,i_{k+1},\dots,i_n\}$ starting from state $s \in S_k$. Moreover, let $\Sigma(s)$ be the set of successors of a state $s$, i.e., those states $s$ reachable through a single transition from $s$. We can now outline our Markov Decision Process.

**Step 0.** (*Initialization*)

$S_0 = \{(T_0,i_0,T_0)\}$. $z_s = 0$ for $s \in S_0$.

```
Step 1. (Forward Computation)
for $k = 1$ to $n$
    Determine the set of feasible states $S_k$.
    for any state $s = (T_k,i'_k,t'_k) \in S_k$ do begin
        Determine the state transition associated with the occurrence of customer $i_k$
        and compute the associated $i''_k$ and $t''_k$.
        If the $k$-th request $i_k$ occurs, compute $\tau_k$.
        Compute $z_s = p_k f_k(\tau_k)$.
    end
end
Set $Z_s = z_s$ for any state $s \in S_n$.

Step 2. (Backward Computation)
for $k = n-1$ to $0$
    for any state $s = (T_k,i''_k,t''_k) \in S_k$ do begin
        Determine the decision associated to state $s$ as the transition from $s$ to state:
        $s'^{(*)} = \text{argmin}_{s' \in \Sigma(s)} Z_{s'}$.
        Then, set $Z_s = Z_{s'^{(*)}}$.
    end
    for any state $s = (T_k,i'_k,t'_k) \in S_k$ do begin
        Determine $Z_s = z_s + p_k Z_{s'} + (1-p_k) Z_{s''}$, where $s'$ and $s''$ are the two decision
        states associated to the occurrence or non-occurrence of the $k$-th request
        provided the vehicle is in state $s$;
    end
end
```

$Z_{(T_0,i_0,T_0)}$ represents the expected cost of an optimal waiting policy.

The number of states is bounded above by $O(n|V|\bar{T})$, where $\bar{T}$ is an upper bound on the service time of customer $i_n$ in an optimal policy (e.g., $\bar{T} = \sum_{r=0}^{n-1} t_{i_r i_{r+1}}$). Hence, the above MDP requires $O(n|V|\bar{T}^2)$ time since $O(\bar{T})$ operations are required for every state.

## 2.5 A numerical example

We now illustrate the above procedures on a numerical example. Let $G(V,A)$ be the graph represented in Figure 2.1, where $V' = \{1,2,3\}$ and $V = \{0\} \cup V' \cup \{a,b,c,d,e\}$. With each vertex in $V'$ are associated the corresponding arrival time and probability, and with each arc is associated its traversal time. Penalties $f_k(\tau_k)$ $(k = 1,2,3)$ are constituted by customers' waiting times, i.e., $f_k(\tau_k) = \tau_k - T_k$, $\tau k \geq T_k$.

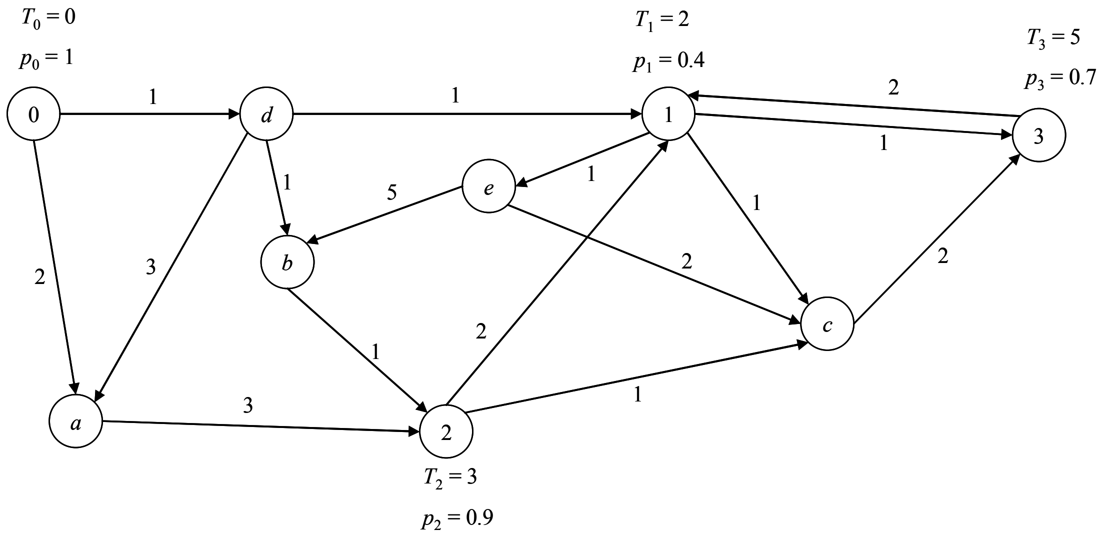


Figure 2.1. Sample network.

Firstly, we compute a lower bound on the expected penalty of an optimal policy. The right-hand side of inequalities (2.3) are:
$L_1 = t_{01}p_0 = 2$
$L_2 = t_{02}(1 - p_1) + (t_{01} + t_{12})p_1 = 1.8 + 3.6 = 5.4$
$L_3 = (t_{01}+t_{12}+t_{23})p_1p_2+(t_{01}+t_{13})p_1(1-p_2)+(t_{02}+t_{23})(1-p_1)p_2+t_{03}(1-p_1)(1-p_2) = 4.32 + 0.12 + 3.24 + 0.18 = 7.86$

Hence, formula (2.4) provides a lower bound equal to:
$LB = 0 + f_1(2)0.4 + f_2(5.4)0.9 + f_3(7.86)0.7 = 0 + 0 \cdot 0.4 + 2.4 \cdot 0.9 + 2.86 \cdot 0.7 = 4.16$

Secondly, we determine an optimal policy through a Markov Decision Process. In Figure 2.2 are reported, for each stage, the associated chance and decision states

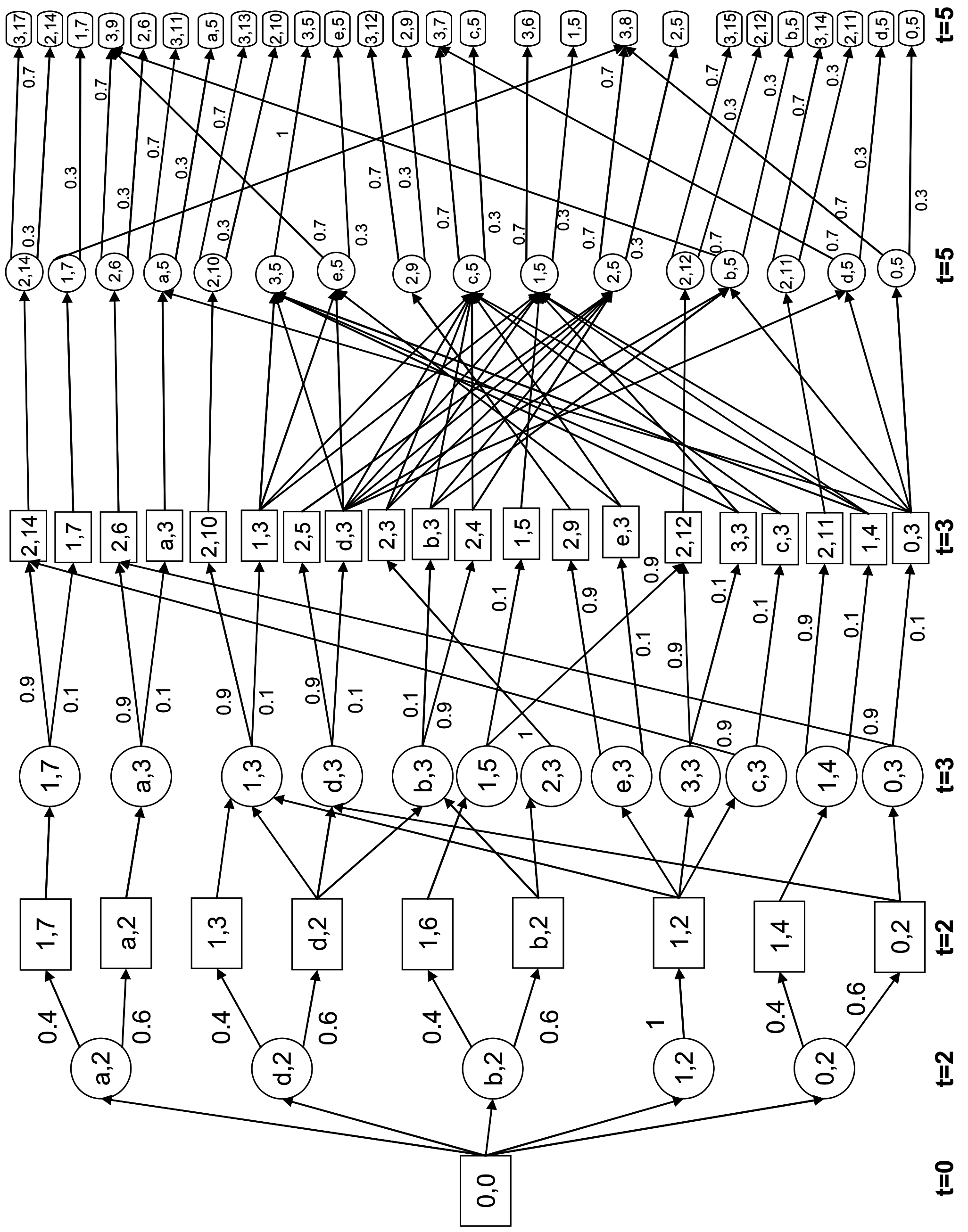


Figure 2.2. State space of the sample problem.

(represented by circles and squares, respectively). Labels $z_s$ and $Z_s$ are shown in Figure 2.3 separated by a semi-colon. The expected cost of an optimal policy is

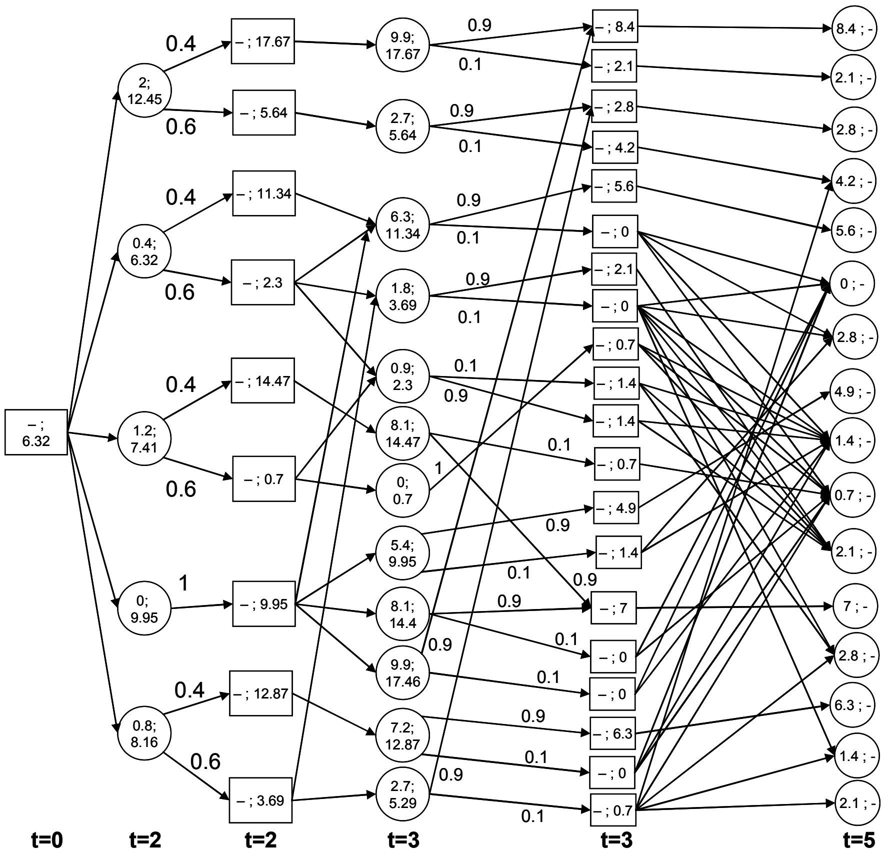


Figure 2.3. Expected penalties in the sample problem.

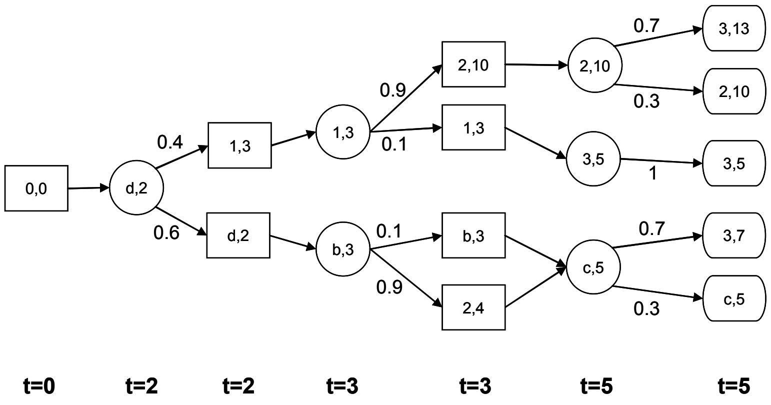


Figure 2.4. Optimal policy for the sample problem.

equal to $Z_{(T_0,i_0,T_0)} = 6.32$. Figure 2.4 illustrates the optimal policy which can be described as follows.

```
At time 0, go empty to vertex $d$. Wait until 2.
if customer 1 requires service then
    go to customer 1 (which is then serviced at time 3); no spare time is left;
    if customer 2 requires service then
        go to customer 2 (which is then serviced at time 10); no spare time is left;
        if customer 3 requires service then
            go to customer 3 (which is then serviced at time 13);
        else
            end of service
        end
    else
        reposition the vehicle to vertex 2 (which it arrives at time 5);
        if customer 3 requires service then
            service this customer (at time 5);
        else
            end of service;
        end
    end
else
    reposition the vehicle to vertex $d$ (where it arrives at time 3);
    if customer 2 requires service then
        go to customer 2 (which is then serviced at time 4);
        reposition the vehicle in vertex c (where it arrives at time 5);
        if customer 3 requires service then
            go to customer 3 (which is then serviced at time 7);
        else
            end of service
        end
    else
        reposition the vehicle to vertex $c$ (where it arrives at time 5);
        if customer 3 requires service then
```

```
            service this customer (at time 7);
        else
            end of service;
        end
    end
end
```

Figures 2.5 and 2.6 depict the Wait-First and Drive-First policies which yield a total expected penalty equal to 9.23 and 9.95, respectively.

## 2.6 Computational Results

In addition, we have solved a number of randomly generated instances on a PC with a Pentium IV processor clocked at 2.8 GHz. Two sets of 50 instances were generated as follows. First, a graph $G(V,A)$ was generated by randomly choosing points in a 100 × 100 square. Then $n(\subseteq |V|)$ customers were chosen at random and an order of visit was determined in a random fashion. In our experiments, we choose $|V| = 50$ and $n = 20,25,30,35,40$. Hence, request occurrence times were chosen as realizations of a Poisson process with $\lambda = 1$ in the first set and $\lambda = 2$ in the second set. Request probabilities were chosen as uniform random numbers in [0,1]. Computational results reported in Table 2.1 indicate that the average lower bound gap is 12.33% for $\lambda = 1$ and 2.42% for $\lambda = 2$. The Markov Decision Process was able to determine the optimal policy always within 1500 seconds for the first set and within 3000 seconds for the second set, while the number of states was always less than 40000 and 60000 for $\lambda = 1$ and $\lambda = 2$, respectively.

The average performance ratio (solution value divided by optimal policy) of the WF heuristic was 1.22 for $\lambda = 1$ and 1.13 for $\lambda = 2$. Similarly, the average performance ratio of the DF heuristic was 1.02 for $\lambda = 1$ and 1.01 for $\lambda = 2$.

Results reported in Table 2.1 show that the instances with $\lambda = 1$ were more difficult to solve, resulting in larger gaps for both the lower and upper bounding techniques and in larger computing times. This can be explained by the fact that larger arrival rates give rise to a busier vehicle. For every test set, the DF policy outperformed the WF policy both in terms of solution quality and computing time.

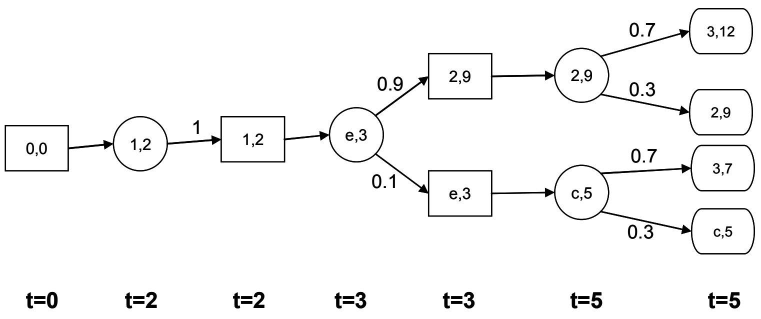


Figure 2.5. Drive-First policy for the sample problem.

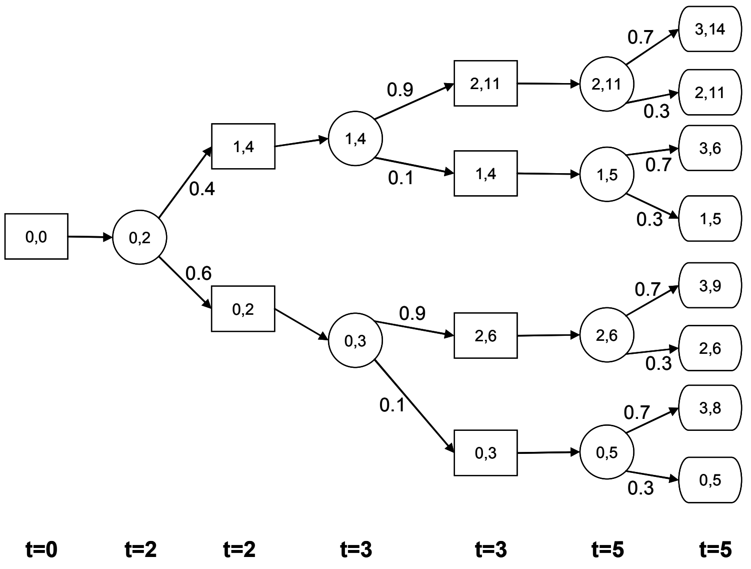


Figure 2.6. Wait-First policy for the sample problem.

## 2.7 Conclusions

In this chapter we have examined exact and heuristic waiting policies for the Dynamic and Stochastic Traveling Salesman Problem under the hypothesis that a probabilistic characterization of the customer requests is available. We have developed a Markov Decision Process as well as a lower bound based on the availability of perfect information. We have assessed the value of two waiting strategies against this lower bound. Our results are based on a number of assumptions that should gradually be removed: a) the hypothesis that request occurrence times $T_1 \leq T_2 \leq \ldots \leq T_n$

Table 2.1. Computational results

| $\lambda$ | $n$ | LB gap (%) | MDP States | MDP time (sec) | WF gap (%) | WF time (sec) | DF gap (%) | DF time (sec) |
|---|---|---|---|---|---|---|---|---|
| | 20 | 13.12 | 8923 | 120.7 | 27.41 | 61.8 | 2.01 | 13.5 |
| | 25 | 19.80 | 12527 | 193.4 | 26.48 | 156.8 | 2.41 | 77 |
| 1 | 30 | 7.62 | 28267 | 753 | 20.13 | 463.8 | 0.95 | 169 |
| | 35 | 2.88 | 33524 | 1075.5 | 15.04 | 807.8 | 1.19 | 457.6 |
| | 40 | 18.22 | 38901 | 1383 | 21.47 | 1135.6 | 2.07 | 858.9 |
| | 20 | 1.67 | 9430 | 118.5 | 14.90 | 58.4 | 0.39 | 17.9 |
| | 25 | 1.17 | 14740 | 222.8 | 13.11 | 171.4 | 0.29 | 78.7 |
| 2 | 30 | 3.96 | 22930 | 431 | 12.34 | 337.4 | 0.28 | 189.9 |
| | 35 | 3.83 | 30211 | 777.5 | 15.97 | 741.5 | 1.50 | 543.4 |
| | 40 | 1.47 | 58968 | 2361.1 | 10.18 | 2262.3 | 0.51 | 1910.1 |

are sorted in non-increasing order; b) the assumption that the order of service is given; c) the hypothesis that a customer request may arise at a single time instant. These extensions are left as a future research. In addition, when removing the over mentioned hypothesis, the MDP will not be handle to handle instances with many customers. Thus, a heuristic will be needed to account for this aspect.

# Chapter 3

# A Priori Routes for the Dynamic Traveling Salesman Problem

## 3.1 Introduction

Advances in information technology and telecommunications, together with continually growing amount of data, offer opportunities for transportation companies to improve both their service offerings and the quality of the service that they provide their customers. One of the most important tools available to companies is the ability to equip drivers with modern tools such as GPS devices, eventually embedded into palmtop computers, in order to exactly know where they are and also to provide them real-time directions. Another key is the ability to communicate in real-time, particularly the ability to communicate new customer requests as they happen. In addition, using the vast of amounts of data that they have stored, transportation companies have detailed statistics about their customers, including their locations and probability distributions on the time of the day that they request service.

Perhaps benefitting most from these advances in technology are delivery companies who emphasize same-day pick-up service. Such companies are often utilized by customers who request service with little or no notice, almost eliminating or, at least, reducing the possibility to construct the entire route in advance. To most efficiently serve these late-requesting customers, delivery companies must develop strategies which account for the late-requesting customers.

In this chapter, we study a dynamic and stochastic routing problem in which we consider a single, uncapacitated vehicle serving a set of known customers locations. The choice to use an uncapacitated vehicle reflects the situation in the courier and package-express industries, where the size of parcels is small enough so that vehicle capacity is not a crucial aspect of the problem. The vehicle begins its route from a known starting point and must complete its journey at a given goal or end node (not necessarily the starting point) by a known time horizon. At the beginning of the time horizon, the driver of the vehicle is aware of a set of customers, called advance-request customers, who have already requested service. These customers may represent packages which are on the vehicle for delivery by the end of the day. In addition to the advance-request customers, there is another subset of customers, called late-request customers, such that, at the beginning of the service horizon, it is unknown whether or not they will require service.

We assume the sets of both advance- and late-request customers as well as their locations are known in advance. We also assume that we know a probability distribution on the likelihood that late-request customers will request service. Our objective is to maximize the expected number of late-request customers who are served. This objective can be seen as equivalent to maximizing customers' Quality of Service (QoS). The impact of customers' QoS on companies is quite important, as low QoS can result in customer reimbursement or lost sales.

In trying to achieve a high-quality QoS, delivery companies face a trade-off. With any realistically-sized number of customers, exact solutions to the problem are impossible. The obvious alternative is to treat the problem entirely dynamically and to, at each decision point, decide which customer to visit next from the pool of advance-request and newly requesting late-request customers. However, this entirely dynamic approach has managerial implications in that it increases the overhead required in driver management as well as affects customer relationships associated with a driver visiting a customer at relatively the same time everyday (Campbell and Thomas, 2007 forthcoming). The alternative is to give drivers an a priori tour of advance-request customers and to insert late-request customers into this a priori tour as they request service. In this chapter, we explore the performance of various a priori routing schemes for use in the dynamic environment previously described. In the remainder of the chapter, we first describe a formal dynamic programming formulation and present the preliminary results. Then, we discuss strategies for implementing a priori routes within this dynamic routing problem as well as our strategies for handling dynamic aspects of the problem. We next outline the experimental design and discuss the results of the computational experiments. Finally, we conclude the chapter and discuss areas for future work.

## 3.2 Model Formulation

In this section, we present a formal dynamic programming formulation for the described dynamic routing problem. The model mirrors the model presented in Thomas (2007), but is repeated here for completeness.

Let $G = (N,E)$ represent the underlying network where $N$ is the set of customers and $E = N \times N$ is the set of arcs connecting customers. For every $n,n' \in N$, we assume that there exists an arc $(n,n') \in E$, including the arc $(n,n)$. That is, the network $G$ is complete. Further, let $t_{ij}$ be the amount of time required to traverse arc $(i,j) \in E$. We can think of the time $t_{ij}$ as deterministic or as the mean travel time on arc $(i,j)$. We assume 1 time unit is required to traverse a self arc. Let $N_J \subseteq N$ be the set of advance-request customers, customers for whom, at the beginning of the time horizon, it is known that service is required. Let $N_I \subseteq N$ be the set of late-request customers, customers such that, at the beginning of the time horizon, it is not known whether or not $n \in N_I$ will require service. We assume that $N_I \cup N_J = N$ and $N_I \cap N_J = \emptyset$. The vehicle begins its route at node $s \in N_J$, the start node, and must complete its journey at a node $\gamma \in N_J$, the goal or end node. We note that $s \neq \gamma$ necessarily.

A decision epoch occurs when the vehicle arrives at a node. Let $t_k$ be a positive integer representing the time of the $k^{th}$ decision epoch, and let $n_k \in N$ be the position of the vehicle at time $t_k$. Let $U = \{0,1,\ldots,T\}$ be the set of possible times when decisions are made, where $T$ is the time after which no more decisions can be made. We can think of $T$ as the time at which the vehicle must have returned to the depot $\gamma$. Let the random variable $K$ represent the total number of decisions and be such that, for a possible realization $\mathcal{K}$ of $K$, $t_{\mathcal{K}} < T$. Our assumptions imply that $t_0 = 0,n_0 = s$, and $\mathcal{K} \leq T$ for every possible realization of $\mathcal{K}$ of $K$.

The random variable $Z^n(t)$ represent the status of customer $n \in N$ at time $t$.

We let

$$Z^n(t) = \begin{cases} 0 & \text{if customer } n \text{ has not yet requested service at time } t \\ 1 & \text{if customer } n \text{ has requested service, but no decision has been made} \\ & \text{as to whether or not to service the customer by time } t \\ 2 & \text{if customer } n \text{ has requested service, has been approved for service,} \\ & \text{but has not been visited by time } t \\ 3 & \text{if customer } n \text{ has been visited} \\ & \text{or if customer } n \text{ requested service and was rejected by time } t \end{cases}$$

for $n = 1, \dots, |N|$, where $|N|$ is the cardinality of $N$. The mechanism used to determine approval for service will be discussed subsequently. Let $Z(t) = \{Z^1(t), \dots, Z^{|N|}(t)\}$, and denote a realization of $Z(t)$ by $z$. Thus, $z \in H = \{0,1,2,3\}^{|N|}$. Further, a realization of $Z^n(t)$, $z^n$, is not equal to 0 for any $n \in N_J$ for any time $t$, and at time $t = 0$, $z^n = 0$ for all $n \in N_I$.

For $z$, a realization of $Z(t)$ for some $t$, define $M(z) = \{n : n \in N, z^n = 2\}$. Then, $M(z)$ is the set of customers who have requested service and been approved by time $t$. We assume that the customers $m_i \in M(z), i = 1, \dots, |M(z(t))|$, for any $t$ and where $|M(z(t))|$ is the cardinality of $M(z)$, are ordered according to some rule or algorithm. In other words, the sequence of customers in $M(z)$ defines a route of approved customers at time $t$. For a realization $z$ of $Z(t = 0)$, we assume that all customers $m \in M(z)$ can be visited and the end node reached by time $T$.

For a realization $z$ of $Z(t)$ for some $t$, let $Q(z) = \{n : z^n = 1\}$. Thus, $Q(z)$ is the set of late-request customers who have requested service, but who have not yet been routed by time $t$. We denote the power set of $Q(z)$ as $2^{Q(z)}$. Then, let $X(q,z), q \in 2^{Q(z)}$, be a function such that:

$$X(q,z) = \begin{cases} 0 & \text{if } q \text{ cannot be feasibly inserted into the route } M(z) \\ 1 & \text{if } q \text{ can be feasibly inserted into the route } M(z). \end{cases}$$

We say that $q \in 2^{Q(z)}$ can be feasibly inserted somewhere into the current route $M(z)$ if every customer $i \in q$ can be inserted into $M(z)$ without the total length of the route exceeding $T$. Finally, define $\bar{2}^{Q(z)} = \{q : q \in 2^{Q(z)}, X(q,z) = 1\}$. That is, $\bar{2}^{Q(z)}$ is the set of subsets of newly arrived customer requests that can feasibly be inserted into the current route $M(z)$. We assume $\emptyset \in \bar{2}^{Q(z)}$.

Define the state space of our problem as $\Omega = \{(n,t,z) : n \in N, t \in U, z \in H\}$. For each $(n,t,z) \in \Omega$, there exists a set of actions $A(n,t,z)$, which are available to the decision maker. At each decision epoch, the decision maker chooses an action from this set. For the problem described here, the decision maker simultaneously controls both insertion and movement with action selection. The insertion actions control which customers are selected for service and thus routed. At any decision epoch and state $(n,t,z)$, $n \neq \gamma$, any set of customers $q \in \bar{2}^{Q(z)}$ can be chosen for insertion. Our assumption that $M(z)$ can be completed before time $T$ for $z$ a realization of $Z(t = 0)$ and the construction of $\bar{2}^{Q(z)}$ imply the vehicle must always reach the end node by time $T$. A movement action controls the motion of the vehicle. For each state $(n,t,z)$, $n \neq \gamma$, the movement actions are to wait at the current location $n$ or to move onto the next node $m_{i+1} \in M(z)$, where $n = m_i$. We assume that, if a subset of customers $q$ is chosen for service, then $M(z)$ is instantaneously augmented and thus $m_{i+1}$ may be such that $m_{i+1} \in q$. Formally, $A(n,t,z) = \{n = m_i, m_{i+1}\} \times \bar{2}^{Q(z)}$. We assume that the selection of movement action $\gamma$ requires the selection of insertion action $\emptyset$. Further, we assume that, for any $t$ and $z$, the only action available in state $(\gamma,t,z)$ is the movement action $\gamma$ and the insertion action $\emptyset$. Subsequent discussion will outline the transitions resulting from action selection.

A decision rule at time $t$ is a function $\delta(\cdot,t,\cdot)$ that selects an available action. Thus, $\delta(n,t,z) \in A(n,t,z)$. A policy is a sequence of decision rules $\pi = \{\delta(n,t,z) : (n,t,z) \in \Omega\}$. We remark that $\delta(\cdot,t,\cdot)$ is implemented only if $t$ is a decision epoch.

For any $n \in N$, $t < T$, and customer status $z^n > 0$, the state dynamics are determined by the selected action. For $n$ such that $z^n = 2$, the selection of the action that includes movement to $m_{i+1}$, where $m_{i+1} = n \in M(z)$, changes the status of customer $n$ from 2 to 3. If the action selected is to wait at the current customer $n$, then the status of customer $n$ remains unchanged. If we choose to insert some number of service requesting customers into the current route, then the status of each $i \in q$, where $q \in \bar{2}^{Q(t)}$, transitions from status 1 to status 2. We assume that this transition is instantaneous.

We assume that $\{Z^i(t), t = 0,1,\ldots,T\}$ and $\{Z^j(t), t = 0,1,\ldots,T\}$ are independent Markov chains for $i \neq j$. Accordingly, for each $n \in N_I$, we assume that the state dynamics for $z^n(t) = 0$ are described by the one-step transition matrix:

$$R_n^{(t,t+1)} = \begin{bmatrix} 1 - \alpha_n^t & \alpha_n^t \\ 0 & 1 \end{bmatrix},$$

where $\alpha_n^t$ is the probability that, between time $t$ and $t + 1$, customer $n$ transitions from $z^n(t) = 0$ to $z^n(t+1) = 1$. We note that, no decisions are made while a vehicle is in transit. A customer requesting service during transit cannot be inserted until the next decision epoch occurs.

Let $P(z' \mid t,z,t')$ be the probability of a transition occurring from $z$ at time $t$ to $z'$ at time $t' > t$. By our assumptions,

$$\begin{aligned} P(z' \mid t,z,t') &= P(Z(t') = z' \mid Z(t) = z) \\ &= \prod_{n=1}^{|N_I|} P(z^n(t') = (z^n)' \mid z^n(t) = z^n), \end{aligned}$$

where each term in the product satisfies an extension of Kolmogorov's equations for the non-stationary case [see Kim, Lewis, and White (2005)] such that

$$R_n^{(t,t')} = \begin{bmatrix} 1-\alpha_n^t & \alpha_n^t \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1-\alpha_n^{t+1} & \alpha_n^{t+1} \\ 0 & 1 \end{bmatrix} \times \cdots \times \begin{bmatrix} 1-\alpha_n^{t'-1} & \alpha_n^{t'-1} \\ 0 & 1 \end{bmatrix}.$$

Our reward structure is described by the function $c(a), a \in A(n,t,z)$, whose value is the number of previously unrouted customers who have requested service and are chosen to have their service request fulfilled. Thus, for an action $a \in A(n,t,z)$, $c(a) = |q|$, where $|q|$ is the cardinality of the set $q \in \bar{2}^{Q(z)}$ identified by the insertion portion of the action. No cost is incurred as the result of travel time or waiting.

Let

$$v^\pi(s,0,z) = E^\pi\left[\sum_{k=0}^{K} c(\delta(n_k,t_k,z_k))\right].$$

be the problem <u>criterion</u> for a policy $\pi$. The problem <u>objective</u> is to find a policy $\pi^\star$, called an <u>optimal policy</u>, such that $v^{\pi^\star}(s,0,z) \geq v^\pi(s,0,z)$ for all policies $\pi$ and for all $z \in \{0,1,2,3\}^{|N|}$.

## 3.3 Preliminary Results and Optimality Equations

All of the results in this section can be found in (Puterman, 1994, Section 4.3). The optimality equation for $t < T$ is

$$v(n,t,z) \;=\; \max\{c(a) + \sum_{z'} P(z' \mid t,z,t')v(n',t',z')) : a \in A(n,t,k)\},$$

where $t' = t + t_{n,n'}$ and $n'$ is the node chosen by the movement portion of the action. The boundary condition is such that $v(\gamma,t,z) = 0$ for all $t$ and $z$. The solution of the optimality equation is unique, and $v(n,t,z) = \max_\pi v^\pi(n,t,z)$, for all $n$,$t$, and $z$. A necessary and sufficient condition for $\pi^\star$ to be optimal is that it is composed of decision rules that cause the maximum in the optimality equation to hold. We refer to $v(n,t,z)$ as the reward or cost-to-go function.

## 3.4 A priori route design

An interesting question to answer is whether or not it is worthwhile to use complex procedures in order to design the a priori route for the advance-request customers. For this purpose, we consider the following approaches for determining the a priori route for our dynamic routing problem. For the computational experiments that follow, we embed these a priori route construction procedures within waiting-strategies approaches for dynamic vehicle routing.

### 3.4.1 Heuristic 1: Shortest Path through Advance-Request Customers (SP Adv)

In our first implementation we construct the route for the advance-request customers in a manner similar to Thomas (2007). More precisely, we set the first advance-request customer as the start node, whereas the last advance-request customer is set as the goal node. We then route the remaining advance-request customers using shortest path through the set of customers. The path is determined through a greedy randomized adaptive search procedure (GRASP) with a savings insertion criterion with a candidate list of size three (Feo and Resende, 1995) and a post-construction greedy local improvement scheme using a 1-shift neighborhood. We then choose $\tau^*$ as the best tour returned after five runs of GRASP.

### 3.4.2 Heuristic 2: Shortest Path through both Advance- and Late-Request Customers (SP All)

This heuristic method builds the a priori route by generating the shortest path through both the advance- and late-request customers. More precisely, we set the first advance-request customer as the start node, whereas the last advance-request customer is set as the goal node. We then route the remaining advance-request customers, along with the late-request customers, using shortest path through the set of customers. The path is determined through a greedy randomized adaptive search procedure (GRASP) with a savings insertion criterion and a post-construction variable neighborhood search improvement scheme using 1-shift and 2-opt as neighborhoods (Mladenović and Hansen, 1997). Let $\tau^*$ be the best found tour after five runs of GRASP. Thus, in order to obtain the a priori route, we simply delete from $\tau^*$ all customers $i$ such that $i \in N_I$.

### 3.4.3 Heuristic 3: Center-of-Gravity Procedure (COG)

Alternatively, we can generate a path through the advance-request customers that maximizes the departure time among all advance-request customers such that the vehicle can finish the remaining advance-request customers and service the center-of-gravity of the late-request customers. This strategy is motivated by the results in Thomas (2007) in which, for a single late-request customer $n$ and a given route $\tau$ of advance-request customers, it is optimal to wait at the location that allows for the latest allowable departure time. Specifically, for a single late-request customer $n$, let $\|n\|_\gamma^j$ be the minimum amount of time required to travel from $j$ to $\gamma$ via $n$ where $n$ is ordered according to some a priori tour of the advance-request customers. That is, $\|n\|_\gamma^j = t_{j,n} + t_{n,j+1} + \sum_{k\in\tau,k=j+1}^{\gamma-1} t_{k,k+1}$. Moreover, let $\bar{t}_j = T - \|n\|_\gamma^j$. We define $\underline{t}_j = \sum_{i=0}^{j-1} t_{i,i+1}$, for each $j \in N_J$, where the index of customers is given by the initial route to serve the advance-request customers. In other words, $\underline{t}_j$ is the earliest time that we can reach customer $j$ given an ordering of advance-request customers.

As shown in Thomas (2007), the result holds for only a single late-request customer, but a heuristic in which the late-request customers are treated as a single aggregated point performs well. Yet, despite its good performance, the heuristic described in Thomas (2007) uses an arbitrary route to order the advance-request customers. Here, we seek a tour $\tau^\star$ such that $\max_{i\in\tau^\star}\{\bar{t}_i\} \geq \max_{i\in\tau}\{\bar{t}_i\}$ for every tour of the advance-request customers $\tau$. In the computation of $\underline{t}_i$ and $\bar{t}_i$, we replace the single late-request customer used in the computations above with the center-of-gravity of all of the late-request customers. See Thomas (2007) for discussion of the aggregation procedure.

To find the tour $\tau^\star$, we use a best-improving local search heuristic using a 1-shift neighborhood.

We initialize our search with the tour $\tau$ which is the shortest path from the start $s$ to the goal $\gamma$ which includes all advance-request customers.

This procedure is embedded into a GRASP, and we choose $\tau^*$ as the best tour returned after five runs of GRASP.

### 3.4.4 Heuristic 4: Probabilistic Traveling Salesman Problem Shortest Path (SP PTSP)

This strategy for constructing the route for the advance-request customers relates our problem to a Probabilistic Traveling Salesman Problem (PTSP) (see, Jaillet, 1985). For each customer $i \in N_I$ we are given a (stationary) probability of a call per unit of time, denoted by $\alpha_i$. Then, for each customer $i$ we compute the $T$-step probability $\alpha_i^T$, that is, the probability that customer $i$ ever calls. Thus we construct a PTSP tour $\tau'$ through all customers in $N$, where $p_i = 1$ if $i \in N_J$, while $p_i = \alpha_i^T$ for $i \in N_I$. Hence, the a priori route $\tau^*$ is obtained as the best tour after five runs of GRASP by removing from $\tau'$ all customers $i$ such that $i \in N_I$.

### 3.4.5 Heuristic 5: A Priori Routing with Sampling and Consensus (SC)

In recent years, sampling procedures combined with consensus have been shown to be a successful in dynamic routing environments (Bent and Van Hentenryck, 2004). In this a priori route construction method, we borrow from these ideas in order to construct an a priori route for the advance-request customers in our dynamic routing problem. In essence, we construct objective maximizing routes for a set of samples and then use a consensus function to determine the best route from the constructed set.

Let $\omega$ represent a set of couples $(i,t)$, $i \in N_I$ and $0 \leq t \leq T$ resulting from a Monte Carlo sample of the set $N_I$. Each couple represents a late-request customer requesting service and the time at which the customer makes the request.

For each sample $\omega$, we construct a route $\tau(\omega)$ which serves each customer $j \in N_J$ as well as as many customers $i$ such that $(i,t) \in \omega$ for some $t$ as is possible. We

note that, because each $i$ is associated with a request time, visits to the sampled late-request customers are constrained by a time window $[t,T-t_{i\gamma}]$ for each $(i,t) \in \omega$. We use $T - t_{i\gamma}$ instead of $T$ in the time window, because if we leave after $T - t_{i\gamma}$, we can't possibly have a feasible tour. Hence, this is a mechanism for tightening the time window. If with each $i$ such that $(i,t) \in \omega$ for some $t$, we associate a reward $r_i$, the problem of maximizing the number of $i$ such that $(i,t) \in \omega$ for some $t$ is then equivalent to a Selective Traveling Salesman Problem (STSP) with time windows and required visits and the closely related Orienteering Problem with Time Windows (OPTW) with required customers (see Laporte and Martello (1990) for a description of the STSP and Kantor and Rosenwein (1992) for a description of the OPTW). However, the constraint requiring visits to all $j \in N_J$ does not appear in the literature. Yet, because our model assumes that there exists a feasible tour which can accommodate all $j \in N_j$, we can essentially relax the constraint into the objective by associating with each $j \in N_J$ a reward $r_j = |\omega| + 1$, where $|\omega|$ is the cardinality of the set $\omega$, whereas we associate with each $i$ such that $(i,t) \in \omega$ for some $t$ a reward $r_i = 1$,. Thus, there is never an optimal tour $\tau(\omega)$ which would include $i$ such that $(i,t) \in \omega$ for some $t$ at the expense of $j \in N_J$. Letting the time window for each $j \in N_J$ be $[0,T]$, we can then apply techniques for the OPTW. In particular, we use the Tree heuristic proposed in Kantor and Rosenwein (1992) to construct $\tau(\omega)$ for each $\omega$. For the sake of completeness, in the following we report verbatim the description given in Kantor and Rosenwein (1992).
Let

- $N = \{1,2,\dots,n\}$ the set of nodes;
- $A = \{(i,j)|i,j \in N\}$ the set of arcs;
- $T_{max}$ the time allotted to complete the path;
- $[e_j,d_j]$ the time window associated with node $j$;
- $t_{ij}$ the travel time from $i$ to $j$;
- $r_j$ the score (reward) of node $j$;
- $y_j = 1$ if node $j$ is visited, 0 otherwise;
- $x_{ij} = 1$ if arc $(i,j)$ is traversed, 0 otherwise;

- $P$ a path;
- $S(P)$ the total score of path $P$;
- $S^*$ the best known total score;
- $k$ the depth of the tree;
- $p(k)$ the $k$-th element in the path;
- $\bar{i}$ the last node added to the path;
- $b_j = \max\{b_i + t_{ij}, e_j\}$ the time at which node $j$ is serviced, where $b_i$ is the time at which node $i$ (the predecessor of node $j$ in the path) is serviced;
- $w_j = \max\{e_j - b_i - t_{ij}, 0\}$ the waiting time at node $j$;
- $0 < \alpha$, $\beta < 1$ parameters used in order to eliminate seemingly unattractive partial paths.

In order to augment a partial path $P$ with node $\bar{j}$ the following rules need to be satisfied:
**R1**: $\bar{j} \notin P$ (a node may exist only once in a path).
**R2**: $b_{\bar{j}} \leq T_{max}$ (path duration constraint).
**R3**: $b_{\bar{j}} \leq d_{\bar{j}}$ (time window constraint).
**R4**: $t_{\bar{i}\bar{j}} \leq V_{max}$ (maximum allowable travel time between two nodes).
**R5**: $w_{\bar{j}} \leq W_{max}$ (maximum allowable waiting time at a node).
**R6**: if $b_{\bar{j}} > \alpha T_{max}$ then $\sum_{j \in P} r_j y_j$ must be $\geq \beta S^*$. We require that $P$ shows sufficient promise of yielding an improved total score over the incumbent best known total score, otherwise it is abandoned.

This heuristic is based upon a depth-first search which is initialized with a partial path $P = \{1\}$ and adds nodes to $P$ until either a complete path is generated or until rules R1-R6 suggest that the current partial path may be abandoned. The best score is updated if a complete path is generated and its total score exceeds the best known score. The formal statement of the algorithm is as follows.

**Step 0.** Initialize $P = \{1\}$, $\bar{i} = 1$, $k = 1$ and $S(P) = r_1 = S^*$.

**Step 1.** Compose a list $\bar{J}_k \subseteq N - n$ of all nodes that satisfy rules R1-R5. All nodes $\bar{j} \in \bar{J}_k$ are initially unmarked.
**Step 2.** Get the next unmarked element $\bar{j} \in \bar{J}_k$. If none exists, go to Step 4.
**Step 3.** $P \leftarrow (P,\bar{j})$. Mark $\bar{j}$. Update the total score of $P$ (i.e., $S(P) = S(P) + r_{\bar{j}}$), the last node added to $P$ (i.e., $\bar{i} = \bar{j}$) and the depth of the tree (i.e., $k = k + 1$). If R6 is satisfied go to Step 1. Else, go to Step 5.
**Step 4.** $P$ may not be augmented with a node $\bar{j} \in \bar{J}_k$. If $P$ may be augmented with node $n$ and if $S(P) + r_n > S^*$, update $S^* = S(P) + r_n$ (i.e., a new path is found that yields an improved total score).
**Step 5.** Backtrack to the previous level of the tree. Update the tree-depth index (i.e., $k = k - 1$), adjust the total score (i.e., $S(P) = S(P) - r_{\bar{j}}$) and update the index to the last node in $P$ (i.e., $\bar{i} = p(k)$).
**Step 6.** Termination test. If $k = 0$, STOP. Else go to Step 2.

Because of the huge computing effort required by such an heuristic in order to find the best path, in order to speed-up the search we modified Step 6 into Step 6' as follows.
**Step 6'**: Termination test. If $k = 0$ OR $P$ contains both the advance-request customers and the late-request customers belonging to the sample OR $P$ contains all the advance-request customers, STOP. Else go to Step 2.

Having taken $\varphi$ samples (in our implementation, $\varphi = 50$) of $N_I$ and constructing a tour from each, we have a set $\Psi$ of tours. Hence, we simply delete from each $\tau \in \Psi$ all customers $i$ such that $i \in N_I$. We call the resulting set of advance-request customer tours $\bar{\Psi}$. We then select our a priori tour from this set through a consensus function. Our choice of function is inspired by the consensus function of Bent and Van Hentenryck (2004), and like that of Bent and Van Hentenryck, represents a least-commitment strategy (see Stefik (1981) for additional examples of least-commitment strategies). Before presenting our consensus function, we first introduce the vector $Y(\tau,\tau')$, where $\tau,\tau' \in \bar{\Psi}$, for which the $i$-th component $Y_i(\tau,\tau') = 1$ if $\tau_i = \tau'_i$ and $Y_i(\tau,\tau') = 0$ otherwise. Then, for each $\tau \in \bar{\Psi}$, we define our consensus function as:

$$f(\tau) = \sum_{\tau' \in \bar{\Psi}, \tau \neq \tau'} \sum_{i=1}^{|N_J|} Y_i(\tau,\tau')$$

We then choose as our a priori tour $\tau^* \in \bar{\Psi}$ such that $f(\tau^*) \geq f(\tau')$ for every $\tau' \in \bar{\Psi}$.

## 3.5 Empirical Results

In this section, we test the effectiveness of each of the proposed a priori routing strategy. To handle the dynamic nature of the problem, we embed the strategies in the dynamic routing framework proposed in Thomas (2007). As discussed in the literature review, waiting strategies are strategies for anticipating dynamic requests. Waiting strategies implicitly or explicitly recognize new service requests will occur and thus choose to have the vehicle wait in anticipation of these new requests. Explicit waiting strategies, as is the strategy used here, choose when and where to wait by exploiting information about both the advance- and late-request customers. In these waiting strategies, newly arriving request for service are accommodated through insertion in an a priori route.

### 3.5.1 Implementation and Data Set Generation

The heuristics discussed in section 3.4 were implemented in C++ using BOOST Graph Library objects (Siek et al., 2001).

As the basis for testing we utilize the eight datasets used in Thomas (2007). The datasets are constructed as follows. Three of the datasets are generated starting from Solomon's 50-customers instances C101, C201, and RC101 datasets (set 1, 2 and 3 in the following) (Solomon, 1987). The other five sets are the 40-customers sets proposed in Dumas et al. (1995) (set 4, 5, 6, 7, 8 in the following). For all eight sets, the time windows are ignored. For each set of customers, we then consider instances having 25% of the customers as late-request and instances having 50% of the customers as late-request. Larsen et al. (2002) note that 25% and 50% percentages for late-request customers correspond to the degrees of dynamism typically found in LTL and package-express problems.

The probabilities that a late-request customer would request service, $\alpha$, are chosen randomly such that the probability that a customer would call over the time horizon of the problem ranged from 0.10 to 0.75. Then, in order to allow for some waiting time along the route, we set the maximum total route time for each instance according to the values used by Thomas (2007). For each dataset, we then consider two values for the horizon. In the manner of Thomas (2007), we refer to these horizons as the 33% and 66% cases. The two cases represent increasing amounts of time available for waiting relative to the time required to service the advance-request customers.

To handle the dynamic part of the problem, we use the LW heuristic as described in Thomas (2007). We use this heuristic as the routing methodology because Thomas (2007) shows that it performs well across a range of parameters. We did not implement other heuristics because the purpose of this chapter is on a priori route design rather than dynamic waiting strategies.

### 3.5.2 Results

This section presents the results of our computational experiments. We compare the performance of the different a priori strategies through simulation. For each dataset and for each a priori strategy, we run the LW heuristic 1,000 times. The average values returned by the heuristics are labeled as SP Adv, SP All, COG, SP PTSP, and CS, for the SP Adv, SP All, COG, SP PTSP, and CS a priori strategies, respectively. Each of the tables presents a series of comparisons. Each comparison compares one of the heuristics to the SP Adv heuristic. This comparison is presented as the percentage less than the mean of the SP Adv solution value returned by the heuristic, or formally, $(100 \times \frac{\text{avg. SP Adv solution - avg. heuristic solution}}{\text{avg. SP Adv solution}})$. These percentage less than the average value in SP Adv are labeled $\Delta$SP All, $\Delta$COG, $\Delta$SP PTSP, and $\Delta$CS for the SP Adv, COG, SP PTSP, and CS heuristics, respectively.

We also compare the heuristics using paired-$t$ confidence interval for the difference of means using the SP Adv heuristic as a standard. For further discussion of the details of this comparison, see Law and Kelton (2000). The comparison between the SP Adv and the SP All heuristics is labeled SP Adv - SP All CI, the comparison between the SP Adv and COG heuristics is labeled SP Adv - COG CI, the comparison between the SP Adv and SP PTSP heuristics is labeled SP Adv - SP PTSP CI, and the comparison between the SP Adv and SC heuristics is labeled SP Adv - SC CI. A confidence interval containing 0 indicates that the performance of the two heuristics is not different at the 95% confidence level. An interval for which the upper and lower confidence limits are positive indicates that the SP Adv heuristic outperformed the alternate heuristic at a 95% confidence level. An interval for which the upper and lower confidence limits are negative indicates that the alternate heuristic outperforms the SP Adv heuristic at the 95% confidence level.

We note that, despite the modified stopping criterion in the SC strategy (which allows shortening the computing times), we are not able to obtain any result for instances 1, 2 and 3 when 50% of the customers are late request. This behavior is the result of the Tree heuristic inability to handle large problem sizes ($n = 50$) and wide time windows (all the planning horizon $[0,T]$).

Table 3.1 shows the results for runs on datasets in which 25% of the customers are late-request and with 33% waiting time. As the data indicate, the SP All strategy heuristic outperforms the SP Adv heuristic in 4 cases out 8, while in other 2 cases the two strategies are equivalent in terms of average values. With respect to the confidence intervals, however, the two heuristics are statistically equivalent in 6 cases out 8.

The performance of the COG heuristic in comparison to SP Adv shows that, by using the COG a priori route, we are able to insert more late-request customers in 5 cases out 8 at the confidence level of 95%, whereas the average result is equivalent for 1 instance.

The performance of the SP PTSP and SP Adv heuristics are indistinguishable at the 95% confidence level in 6 out of 8 cases. Moreover, instance 2 shows better performance of SP PTSP, whereas in the remaining case SP Adv outperforms SP PTSP.

Finally, the SC heuristic performs particularly poorly. The results show the SC heuristics on average 23% to 119% worse than the SP Adv heuristic. We speculate that this poor performance is in part due to the heuristic used to solve the OPTW. To maintain reasonable computation times, the heuristic was often prematurely terminated and thus returning poor solutions.

For the case of 25% of late-request customers and with 33% waiting time, it is instructive to consider an example of the a priori tours returned by each of the heuristics. Figure 3.1 presents graphical representation of the a priori routes obtained with each heuristic for dataset 4. What is evident from the observation of the figures is that COG heuristic (Figure 3.1(c)) produces essentially the same route as SP Adv heuristic (Figure 3.1(a)). This result occurs because COG seeks to maximize the departure time. It is reasonable that the maximization of departure time is given by the tour that is shortest, in our case the tour of SP Adv. At the same time, SP All heuristic (Figure 3.1(b)) shows the same behavior as SP PTSP strategy (Figure 3.1(d)), mainly because the placement of the late-request customers and the fact that the advance-request customers occur with certainty leads to identical tours. The SC tour differs from the other tours. However, the fact that the tour crosses itself is not completely unexpected, as TSPTW tours as well as Probabilistic Traveling Salesman Problem with Deadlines (PTSPD) tours tend to do that.

Table 3.2 shows the results for runs on datasets in which 25% of the customers are late-request and with 66% waiting time. The results are analogous to those in Table 3.1; no strategy clearly outperforms all others.

Table 3.3 shows the results for runs on datasets in which 50% of the customers are late-request and with 33% waiting time. As the data indicate, when the number of late-request customers grows, the SP Adv strategy heuristic performs quite poorly if compared to the other strategies.

With respect to the comparison between SP Adv and SP All, the second strategy outperforms the first at the 95% confidence level in all cases but one, with average improvements ranging between 1% and 12%.

In comparison to the SP Adv heuristic, the COG heuristic provides improved results in 5 cases out 8, whereas the average result is statistically equivalent for 1 instance.

The SP PTSP heuristic also outperforms the SP Adv strategy, with better results in 6 cases out 8 at a confidence level of 95%.

The SC strategy is still not competitive, although it does outperform the SP Adv heuristic in 2 of 5 instances at the 95% confidence level. The SP Adv heuristic outperforms the SC in 2 cases with the same confidence level.

In order to identify a best strategy, we conduct a pairwise comparison of the SP All, COG, and SP PTSP strategies. This comparison is reported in Table 3.4. We compare the percentage less than the mean of each strategy solution value returned by each other heuristic, or formally, $(100 \times \frac{\text{avg. Heuristic x solution - avg. Heuristic y solution}}{\text{avg. Heuristic x solution}})$. These percentage less than the average value are labeled $\Delta$SP All - COG, $\Delta$SP All - SP PTSP, and $\Delta$COG - SP PTSP for the SP All and COG, SP ALL and SP PTSP, and COG and SP PTSP, respectively.

We also compare the heuristics using paired-*t* confidence interval for the difference of means using the SP Adv heuristic as a standard. The comparison between the SP All and the COG heuristics is labeled SP All - COG CI, the comparison between the SP All and SP PTSP heuristics is labeled SP All - SP PTSP CI, and the comparison between the COG and SP PTSP heuristics is labeled COG - SP PTSP CI.

The comparison between SP All and COG points out that the former strategy is better than the latter in 4 cases out 8, whereas in all the other cases the two heuristics perform approximately the same at the 95% confidence level.

Moreover, if we compare the SP All and the SP PTSP heuristics, the former strategy outperforms the latter in 2 cases out of 8. The heuristics are statistically indistinguishable in the other instances.

Comparing the COG and the SP PTSP strategies there is not a clear evidence of a strategy outperforming the other, given that the SP PTSP is better in 2 cases, the COG is better in only 1 case, whereas the other cases do not allow any statistically significant conclusion.

To better understand the results of the 50% of late-request customers and with 33% waiting time case, Figure 3.2 presents a graphical representation of the a priori routes obtained with each heuristic on dataset 4. From the figure, we can clearly see that the structure of the tours diverges as the the percentage of late-request customers has increased. We note that the SP All tour cuts across the middle of the $(x,y)$ plane accounting for the late-request customers in that middle region, whereas the SP Adv tour ignores this area of the region. The advantage of the COG and PTSP heuristics follows similar reasoning.

## 3.6 Conclusions

In this chapter we have studied a dynamic and stochastic routing problem in which a single, uncapacitated vehicle has to serve a set of known customers locations but in which some customers request service while the vehicle is en route.

We have devised a set of a priori routing schemes for use in such a dynamic environment. We have first described a formal dynamic programming formulation and presented the preliminary results. Then, we have discussed strategies for implementing a priori routes within this dynamic routing problem. We have next outlined the experimental design and discussed the results of the computational experiments.

Results reported in Tables 3.1 and 3.2 indicate that there is no value in devising involved strategies for the construction of the a priori routes when the percentage of late-request customers is low. Indeed, a simple strategy like SP Adv performs comparable to more involved heuristics. On the other hand, Table 3.3 shows that, when the number of late-request customers increases to 50%, more involved procedures offer improvement. In this case, a good choice for an a priori tour construction heuristic is the SP All heuristic. It performs comparably to the other heuristics, in some cases better, and offers a straightforward implementation.

In future work, we are interested in whether or not it is possible to construct good a priori routes such that, using waiting strategies, we are able to benefit from the ease of managing a priori routes without sacrificing the cost savings available in purely dynamic strategies.

We are also interested in finding an efficient heuristic for the OPTW. The SC heuristic is intuitively appealing, but requires an improved solution approach to truly assess its value.

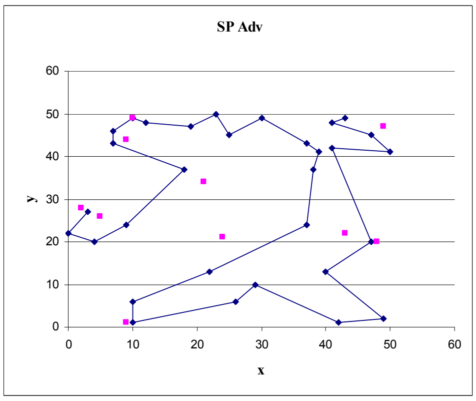
Table 3.1. Results for 25% Late-Request Customers and 33% Waiting Time.

(a) Comparisons (SP Adv - SP All) and (SP Adv - COG)

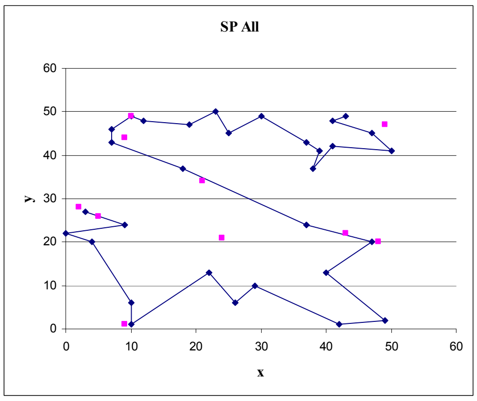
| Set | SP Adv | SP All | *Δ* SP All | SP Adv - SP All CI | COG | *Δ* COG | SP Adv - COG CI |
|---|---|---|---|---|---|---|---|
| 1 | 9.27 | 9.34 | -1% | (-0.18, 0.03) | 9.32 | -1% | (-0.16, 0.06) |
| 2 | 9.97 | 10.40 | -4% | (-0.52, 0.35) | 10.28 | -3% | (-0.41, -0.23) |
| 3 | 9.61 | 9.80 | -2% | (-0.30, -0.09) | 9.73 | -1% | (-0.22, -0.21) |
| 4 | 7.76 | 7.77 | 0% | (-0.10, 0.09) | 7.86 | -1% | (-0.19, -0.01) |
| 5 | 7.64 | 7.59 | 1% | (-0.05, 0.15) | 7.55 | 1% | (-0.01, 0.19) |
| 6 | 8.04 | 8.10 | -1% | (-0.14, 0.04) | 8.07 | 0% | (-0.12, 0.06) |
| 7 | 7.64 | 7.38 | 4% | (0.16, 0.36) | 7.09 | 8% | (0.45, 0.60) |
| 8 | 7.88 | 7.88 | 0% | (-0.10, 0.09) | 7.94 | -1% | (-0.16, 0.04) |

(b) Comparisons (SP Adv - SP PTSP) and (SP Adv - SC)

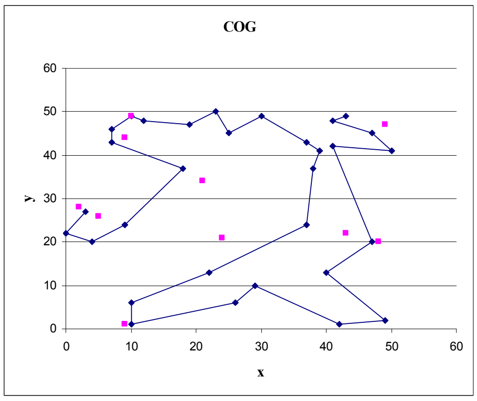
| Set | SP Adv | SP PTSP | *Δ* SP PTSP | SP Adv - SP PTSP CI | SC | *Δ* SC | SP Adv - SC CI |
|---|---|---|---|---|---|---|---|
| 1 | 9.27 | 9.32 | -1% | (-0.16, 0.06) | | | |
| 2 | 9.97 | 10.33 | -3% | (-0.45, -0.27) | | | |
| 3 | 9.61 | 9.65 | 0% | (-0.14, 0.07) | | | |
| 4 | 7.76 | 7.71 | 1% | (-0.05, 0.14) | 6.12 | 27% | (1.53, 1.75) |
| 5 | 7.64 | 7.58 | 1% | (-0.04, 0.16) | 6.21 | 23% | (1.32, 1.53) |
| 6 | 8.04 | 8.06 | 0% | (-0.11, 0.07) | 5.88 | 37% | (2.06, 2.26) |
| 7 | 7.64 | 7.52 | 2% | (0.02, 0.22) | 3.48 | 119% | (4.05, 4.26) |
| 8 | 7.88 | 7.93 | -1% | (-0.14, 0.05) | 5.84 | 35% | (1.93, 2.14) |



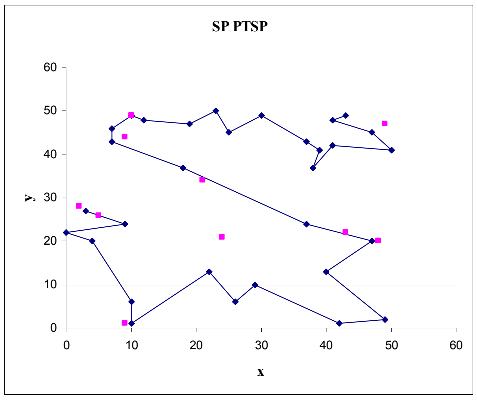
(a)



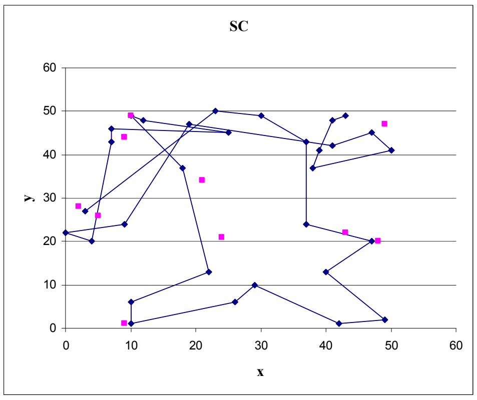
(b)



(c)



(d)



(e)

Figure 3.1. Initial routes for the five a priori heuristics for set 4 in the case with 25% Late-Request Customers and 33% Waiting Time. Squared points represent the late-request customers.

Table 3.2. Results for 25% Late-Request Customers and 66% Waiting Time.

(a) Comparisons (SP Adv - SP PTSP) and (SP Adv - SC)

| Set | SP Adv | SP All | Δ SP All | SP Adv - SP All CI | COG | Δ COG | SP Adv - COG CI |
|---|---|---|---|---|---|---|---|
| 1 | 9.91 | 9.89 | 0% | (-0.07, 0.12) | 9.94 | 0% | (-0.13, 0.07) |
| 2 | 10.77 | 10.74 | 0% | (-0.05, 0.11) | 10.72 | 0% | (-0.03, 0.13) |
| 3 | 10.51 | 10.58 | -1% | (-0.15, 0.01) | 10.62 | -1% | (-0.19, -0.03) |
| 4 | 8.38 | 8.30 | 1% | (-0.01, 0.17) | 8.31 | 1% | (-0.01, 0.16) |
| 5 | 8.21 | 8.22 | 0% | (-0.10, 0.08) | 8.21 | 0% | (-0.10, 0.08) |
| 6 | 8.58 | 8.56 | 0% | (-0.06, 0.11) | 8.53 | 1% | (-0.04, 0.14) |
| 7 | 8.09 | 8.16 | -1% | (-0.16, 0.02) | 8.11 | 0% | (-0.10, 0.08) |
| 8 | 8.41 | 8.38 | 0% | (-0.05, 0.11) | 8.42 | 0% | (-0.10, 0.08) |

(b) Comparisons (SP Adv - SP PTSP) and (SP Adv - SC)

| Set | SP Adv | SP PTSP | Δ SP PTSP | SP Adv - SP PTSP CI | SC | Δ SC | SP Adv - SC CI |
|---|---|---|---|---|---|---|---|
| 1 | 9.91 | 9.90 | 0% | (-0.08, 0.11) | | | |
| 2 | 10.77 | 10.75 | 0% | (-0.06, 0.09) | | | |
| 3 | 10.51 | 10.61 | -1% | (-0.18, -0.02) | | | |
| 4 | 8.38 | 8.33 | 1% | (-0.04, 0.14) | 8.10 | 4% | (0.20, 0.37) |
| 5 | 8.21 | 8.15 | 0% | (-0.06, 0.13) | 7.93 | 3% | (0.18, 0.37) |
| 6 | 8.58 | 8.56 | 0% | (-0.06, 0.11) | 7.94 | 8% | (0.54, 0.73) |
| 7 | 8.09 | 8.18 | -1% | (-0.18, 0.01) | 7.14 | 13% | (0.85, 1.04) |
| 8 | 8.41 | 8.38 | 0% | (-0.05, 0.12) | 8.26 | 2% | (0.07, 0.24) |

Table 3.3. Results for 50% Late-Request Customers and 33% Waiting Time.

(a) Comparisons (SP Adv - SP PTSP) and (SP Adv - SC)

| Set | SP Adv | SP All | $\Delta$ SP All | SP Adv - SP All CI | COG | $\Delta$ COG | SP Adv - COG CI |
|---|---|---|---|---|---|---|---|
| 1 | 18.58 | 19.81 | -6% | (-1.42, -1.04) | 19.25 | -3% | (-0.87, -0.47) |
| 2 | 21.42 | 21.92 | -2% | (-0.62, -0.38) | 21.37 | 0% | (-0.09, 0.17) |
| 3 | 22.66 | 21.37 | 6% | (1.16, 1.42) | 21.20 | 7% | (1.32, 1.60) |
| 4 | 15.85 | 16.11 | -2% | (-0.40, -0.12) | 16.15 | -2% | (-0.44, -0.15) |
| 5 | 14.75 | 15.78 | -7% | (-1.17, -0.89) | 15.67 | -6% | (-1.06, -0.77) |
| 6 | 15.07 | 17.05 | -12% | (-2.13, -1.83) | 16.83 | -11% | (-1.92, -1.61) |
| 7 | 13.71 | 15.22 | -10% | (-1.66, -1.35) | 15.17 | -10% | (-1.60, -1.30) |
| 8 | 15.08 | 15.30 | -1% | (-0.36, -0.07) | 15.08 | 0% | (-0.15, 0.17) |

(b) Comparisons (SP Adv - SP PTSP) and (SP Adv - SC)

| Set | SP Adv | SP PTSP | $\Delta$ SP PTSP | SP Adv - SP PTSP CI | SC | $\Delta$ SC | SP Adv - SC CI |
|---|---|---|---|---|---|---|---|
| 1 | 18.58 | 19.82 | -6% | (-1.44, -1.05) | | | |
| 2 | 21.42 | 21.57 | -1% | (-0.28, -0.03) | | | |
| 3 | 22.66 | 20.93 | 8% | (1.56, 1.90) | | | |
| 4 | 15.85 | 16.13 | -2% | (-0.42, -0.13 ) | 16.01 | -1% | (-0.31, -0.01) |
| 5 | 14.75 | 15.54 | -5% | (-0.94, -0.64) | 13.61 | 8% | (0.98, 1.32) |
| 6 | 15.07 | 17.11 | -12% | (-2.18, -1.90) | 16.55 | -9% | (-1.64, -1.32) |
| 7 | 13.71 | 15.10 | -9% | (-1.54, -1.22) | 13.61 | 1% | (-0.07, 0.27) |
| 8 | 15.08 | 15.34 | -2% | (-0.40, -0.11) | 10.94 | 38% | (3.99, 4.28) |

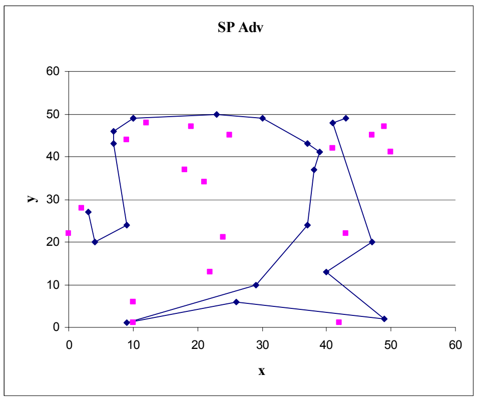
Table 3.4. Results for 50% Late-Request Customers and 33% Waiting Time.

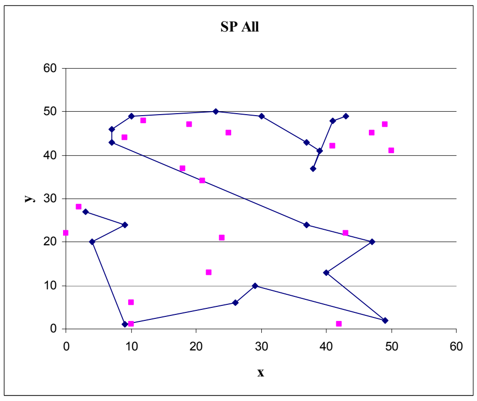
(a) Comparisons (SP All - COG) and (SP All - SP PTSP)

| Set | $\Delta$ SP All - COG | SP All - COG CI | $\Delta$ SP All - SP PTSP | SP All - SP PTSP CI |
|---|---|---|---|---|
| 1 | 3% | (0.39, 0.73) | 0% | (-0.18, 0.15) |
| 2 | 3% | (0.42, 0.67) | 2% | (0.23, 0.47) |
| 3 | 1% | (0.02, 0.32) | 2% | (0.26, 0.63) |
| 4 | 0% | (-0.17, 0.09) | 0% | (-0.15, 0.11) |
| 5 | 1% | (-0.02, 0.24) | 2% | (0.11, 0.38) |
| 6 | 1% | (0.09, 0.33) | 0% | (-0.18, 0.06) |
| 7 | 0% | (-0.09, 0.20) | 1% | (-0.02, 0.28) |
| 8 | 2% | (0.08, 0.38) | 0% | (-0.17, 0.09) |

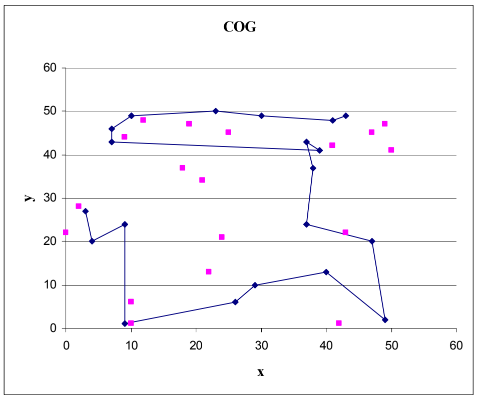
(b) Comparison (COG - SP PTSP)

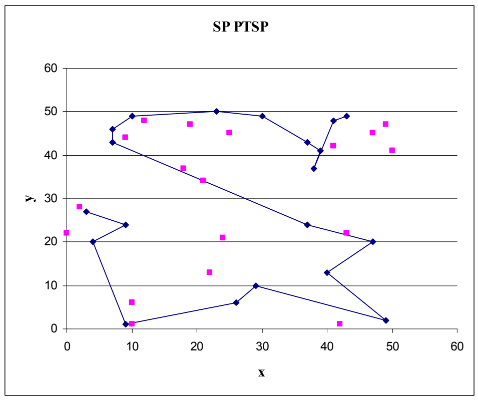
| Set | $\Delta$ COG - SP PTSP | COG- SP PTSP CI |
|---|---|---|
| 1 | -3% | (-0.74, -0.41) |
| 2 | -1% | (-0.32, -0.07) |
| 3 | 1% | (0.08, 0.46) |
| 4 | 0% | (-0.11, 0.15) |
| 5 | 1% | (-0.01, 0.26) |
| 6 | -2% | (-0.39, -0.16) |
| 7 | 0% | (-0.08, 0.23) |
| 8 | -2% | (-0.41, -0.12) |



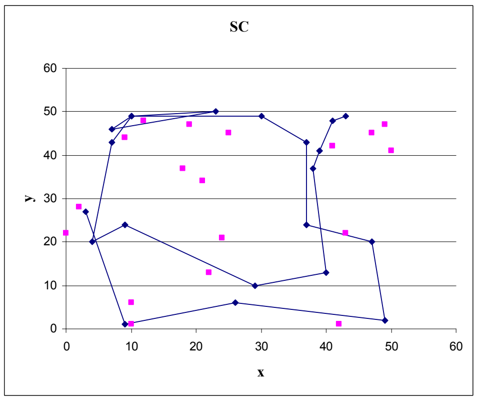


(a) (b)





(c) (d)



(e)

Figure 3.2. Initial routes for the five a priori heuristics for set 4 in the case with 50% Late-Request Customers and 33% Waiting Time. Squared points represent the late-request customers.

# Chapter 4

# The Dynamic and Stochastic Vehicle Dispatching Problem with Pickups and Deliveries

## 4.1 Introduction

Same-day couriers are utilized by clients who require maximum speed and security for deliveries of letters and small parcels in urban areas. Customers usually request couriers with little or no notice, all but eliminating the ability to construct routes and schedules in advance. The traditional model of same-day courier service utilizes human controllers who communicate with bicycle, motorcycle, car and van couriers via radio or mobile phone. Controllers ask couriers to relay their location information and then assign jobs to the most appropriate vehicle. This model is not only inefficient, but also suffers from errors inherent with the involvement of a human element. Indeed, the recent advances in communication and information technologies, that now allow data on courier locations and customers requests to be obtained and processed in real-time, have stimulated research on algorithms for dynamic vehicle routing and dispatching problems. Unlike their static counterparts, these problems are characterized by data which are disclosed in a dynamic fashion over a planning horizon. In this context, decisions made at an early stage of the planning horizon might affect the ability to make good decisions at a later stage. Nonetheless, large part of the current literature is focused on algorithms reacting to new requests only once they have occurred, while neglecting available stochastic information. As a result, these algorithms are not able to take advantage of recurrent patterns in customer demands and vehicle travel times, which usually constitutes an easy task for human controllers.

The purpose of this chapter is to describe and assess anticipatory heuristics for the dynamic and stochastic *Vehicle Dispatching Problem with Pickups and Deliveries* that anticipate future demands through a Monte Carlo sampling procedure. The insight gained by simulating near-future demand is used in order to manage in an unified way several kinds of decisions, including vehicle dispatching, route scheduling and idle vehicle relocation.

## 4.2 Problem Statement

The dynamic and stochastic *Vehicle Dispatching Problem with Pickups and Deliveries* (VDPPD) is defined on a graph $G = (V,A)$, where $V$ is a vertex set and $A$ is an arc set. A fleet of $m$ vehicles, located at a depot $i_0 \in V$ at time $t = 0$, has to service a number of pick-up and delivery requests $\{(i_k^+,i_k^-,T_k) : k = 1,2,...\}$, where $i_k^+ \in V$, $i_k^- \in V$, $T_k \geq 0$ are respectively, the pickup point, the delivery point and the occurrence time of the $k$-th request. Vertices may represent individual customer locations or the zones in which the service territory is divided. For an extensive treatment of diversion issues in real-time vehicle dispatching the reader is referred to Regan et al. (1994, 1995) and Ichoua et al. (2000). Let $t_{ij}$ be the shortest travel time from vertex $i \in V$ to vertex $j \in V$. As is common in the same-day courier industry, the aim is to maximize the overall customer service level rather than minimize the total traveled distance. Let $\tau_k$ be the delivery time of the $k$-th request. To each customer is associated a non-decreasing and convex penalty function $f_k(\tau_k)$ expressing the inconvenience associated with customer $i_k$. This definition includes the case in which $f_k(\tau_k)$ represents the customer *waiting time* (i.e., $f_k(\tau_k) = \tau_k - T_k$, $\tau_k \geq T_k$) or a more involved penalty function (e.g., $f_k(\tau_k) = 0$, $T_k \leq \tau_k \leq D_k$ and $f_k(\tau_k) = \tau_k - D_k$, $\tau_k \geq D_k$, where $D_k$ is a *soft deadline* associated with the $k$-th request).

The static version of the VDPPD amounts to determining an ordered sequence of locations on each vehicle route such that:

1. each route starts at the depot;
2. a pickup and its associated delivery are satisfied by the same vehicle;
3. a pickup is always made before its associated delivery;
4. the total penalty incurred by the vehicles $z = \sum_k f_k(\tau_k)$ is minimized.

In the dynamic variant, we also have to adequately distribute waiting time along the routes, since this may affect the overall solution quality, as well as reposition idle vehicles to anticipate future demand. In this chapter we assume that the requests arrive according to a known stochastic process. The objective function to be minimized is the expected customer inconvenience over the planning horizon:

$$z = \sum_k E[f_k(\tau_k)]$$

where $E[f_k(\tau_k)]$ is the expected penalty associated to the delivery of the $k$-th request. Moreover, we assume that a vehicle cannot be diverted away from its current destination to service a new request in the vicinity of its current position. Such an opportunity can be exploited when communication between the dispatch office and the couriers can take place at any time, which is usually not the case of bike and motorbike drivers which is the focus of this chapter.

In this chapter we develop a mechanism in which, any time a new request arrives, the short-term arrival process is sampled and alternative solutions are compared through a fully sequential procedure for indifference zone selection (Kim and Nelson, 2001). This approach allows determining the number of samples required for each alternative solution in order to select the best insertion with a given level of confidence.

## 4.3 The anticipatory algorithms

We have devised the following anticipatory mechanism which we have then embedded in both an insertion and a local search procedure.

Let $P_k \subseteq \{1, \ldots, k\}$ be the set of pending requests (i.e., the requests occurred but not yet serviced) at time $T_k$, when request $(i_k^+, i_k^-, T_k)$ arrives. A *reactive* algorithm generates a new solution incorporating $i_k^+$ and $i_k^-$ with the aim to minimize the total inconvenience associated to requests $P_k$:

$$z_k = \sum_{r \in P_k} f_r(\tau_r).$$

On the contrary, our *anticipatory* algorithms aim at minimizing the sum of $z_k$ plus the expected value (under perfect information) of the total penalty $\xi^{[t_k,t_k+\Delta t_k]}$ associated to the requests arriving in the short term future $[t_k,t_k+\Delta t_k]$:

$$z'_k = \sum_{r \in P_k} f_r(\tau_r) + E[\xi^{[t_k,t_k+\Delta t_k]}], \tag{4.1}$$

where $\Delta t_k$ is the short term duration. Of course, our procedures become reactive if $\Delta t_k = 0$ $(k = 1,2,\ldots)$.

We now provide an in-depth description of the main ingredients of the two anticipatory algorithms.

## 4.3.1 Approximation of the near-future inconvenience

In order to estimate $E[\xi^{[t_k,t_k+\Delta t_k]}](s)$ for any given solution $s$, we generate $n_s$ samples of the near-future demand and compute the corresponding total penalties under perfect information: $\xi^{[t_k,t_k+\Delta t_k]}(s)$ $(j = 1,\ldots,n_s)$. Then, we approximate $E[\xi^{[t_k,t_k+\Delta t_k]}](s)$ through its sample mean:

$$\overline{\xi^{[t_k,t_k+\Delta t_k]}(s)} = \frac{\sum_{j=1}^{n_s} \xi_j^{[t_k,t_k+\Delta t_k]}(s)}{n_s}$$

as well as compute confidence intervals on $E[\xi^{[t_k,t_k+\Delta t_k]}](s)$.

For every solution $s$ and for every sample $j$ $(j = 1,\ldots,n_s)$, we approximate the penalties under perfect information $\xi^{[t_k,t_k+\Delta t_k]}(s)$ by means of a cheapest insertion heuristic.

## 4.3.2 Determination of the number of samples

In a simulation-based optimization setting, the evaluation of the objective function $z'$ via discrete event simulation sets up a dichotomy not present in deterministic optimization: that of the search process versus the evaluation process (Fu, 2002). Indeed, most of the computation is spent estimating the objective function for given values of the decision variables, a reversal of the deterministic setting, where the search is the primary computational burden. Our approach is based on the observation that there is no reason a priori to assume that the number of samples $n_s$ should be the same for all solutions $s$. Indeed, our procedure compares the alternative solutions by using a *fully sequential Indifference Zone Selection* (IZS) procedure (Kim and Nelson, 2001). Such a procedure requires that the user specifies two parameters:

- an indifference zone width $\delta$;
- a confidence level $1 - \alpha$.

The goal of the IZS procedure is to select a solution with expected penalty that is within $\delta$ units of the optimal performance with a given level of confidence $1 - \alpha$. In practice, parameter $\delta$ is set equal to the smallest relevant absolute difference in the expected penalty. An objective function gap less than $\delta$ units is considered negligible. The procedure takes only a single basic output from each alternative still in contention at each stage. Also, if there exists clear evidence that a solution is inferior, then it will be immediately eliminated from consideration. We now illustrate this concept through an example (Figure 4.1). For each of the five alternative

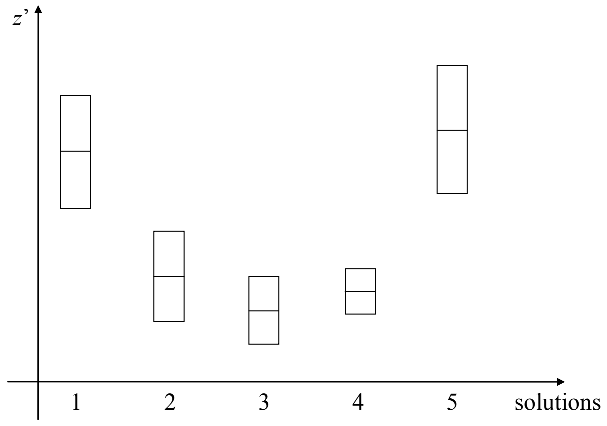


Figure 4.1. Boxplot representation of five alternative solutions.

solutions, 10 samples are initially taken and 99% confidence intervals are computed.

On the basis of this results, we can discard solutions 1 and 5. We then take one more sample for each of the three remaining solutions and compare the updated confidence intervals. Such a procedure is iterated until all solutions but one are eliminated.
As customary in simulation-based optimization, we make use of common random numbers (CRNs) in order to reduce variance (Kim and Nelson, 2006). A more formal description of the screening and selection procedure is as follows.

**Step 1.** (*Initialization.*) Select a confidence level $1-\alpha$, an indifference zone parameter $\delta$ and a first stage sample size $n_0 \geq 2$. Let $S = \{1,\dots,p\}$ be the set of solutions still in contention. Set

$$\eta = \frac{1}{2}\left[\left(\frac{2\alpha}{p-1}\right)^{-2/(n_0-1)} - 1\right].$$

Obtain $n_0$ independent outputs $\xi_l^{[t_k,t_k+\Delta t_k]}(s)$ $(l = 1,\dots,n_0)$ from each solution $s \in S$. Set $r = n_0$.

**Step 2.** (*Screening.*) Let $\overline{\xi_s(r)} = \frac{\sum_{l=1}^{r} \xi_l^{[t_k,t_k+\Delta t_k]}(s)}{r}$ denote the sample mean of the first $r$ outputs from solution $s$. For all $s' \neq s$ compute $\widehat{\sigma}^2_{ss'}(r)$, the sample variance of the difference between solutions $s$ and $s'$:

$$\widehat{\sigma}^2_{ss'}(r) = \frac{\sum_{l=1}^{r}\left(\xi_l^{[t_k,t_k+\Delta t_k]}(s) - \xi_l^{[t_k,t_k+\Delta t_k]}(s') - \left[\overline{\xi}_s(r) - \overline{\xi}_{s'}(r)\right]\right)^2}{r-1}.$$

Set $S^{old} = S$. Let

$$S = \{s : s \in S^{old} \quad \text{and} \quad \overline{\xi}_s(r) \leq \overline{\xi}_{s'}(r) + W_{ss'}(r), s' \in S^{old}, s' \neq s\}$$

where

$$W_{ss'}(r) = \max\left\{0, \frac{\delta}{2r}\left(\frac{h^2\widehat{\sigma}^2_{ss'}(r)}{\delta^2} - r\right)\right\},$$

and $h^2 = 2\eta(r-1)$. $W_{ss'}(r)$ denotes how far the sample mean from solution $s$ can drop below the sample means of the other solutions without being eliminated.

**Step 3.** (*Stopping Rule.*) If $|S| = 1$, then stop and select the solution whose index is in $S$ as the best. Otherwise, take one additional output $\xi_{r+1}^{[t_k,t_k+\Delta t_k]}(s)$ from each solution $s \in S$, set $r = r + 1$ and go to Step 2.

### 4.3.3 Generation of alternative solutions

In the anticipatory insertion procedure, we generate alternative solutions as follows. At any time, the current solution can be seen as a set of routes $R_j = \{(i_{0,j},i_{1,j},\dots,i_{n_j,j},i_{n_j+1,j}),\tau_{0,j},\tau_{1,j},\dots,\tau_{n_j,j}\}$ $(j = 1,\dots,m)$, where $i_{0,j}$ is the current position of vehicle $j$, $i_{1,j},\dots,i_{n_j,j}$ are pickup and delivery points to be visited by vehicle $j$, $i_{n_j+1,j}$ is a parking position, and $\tau_{0,j},\tau_{1,j},\dots,\tau_{n_j,j}$ are the waiting times of vehicle $j$ at $i_{0,j},i_{1,j},\dots,i_{n_j,j}$, respectively. In order to keep the computational effort within reasonable limits, it is worth selecting a reduced number $\kappa$ of promising solutions to be assessed through near-future simulation. When a request $(i_k^+,i_k^-,T_k)$ occurs, we generate every feasible insertion. Then, for each of the best $\kappa$ insertions with respect to $z_k$, we enumerate all the feasible solutions associated to those insertions (by suitably inserting waiting times along the individual route receiving request $k$ as well as repositioning the corresponding vehicle when it becomes idle). In order to make the solutions feasible in the real-world, we impose that the total waiting time along a route does not exceed a maximum amount $MWT$ as well as we require the waiting time at any node to be a multiple of a *quantum* $WQ$. Moreover, we assume that idle vehicles may be relocated only at a restricted number of home positions $H \subseteq V$ (which is what happens in real world). For instance, if $MWT = 20$ minutes, $WQ = 10$ minutes and $H = \{i_q,i_{q'}\}$, a customer sequencing $(i_1^-,i_3^+,i_2^-,i_3^-)$ gives rise to the route schedules reported in Table 4.1. It is worth noting that two distinct route schedules, which coincide during the short term future $[t_k,t_k + \Delta t_k]$, incur the same expected penalty (4.1) and require a single simulation experiment.

### 4.3.4 Anticipatory local search

In our local search scheme, the neighbors of a given solution are obtained by removing in turn each pending request from its current route and reinserting it feasibly through the anticipatory insertion procedure described before. In order to keep the computational effort into reasonable limits, at each iteration we remove only the pending requests currently allocated to the routes modified in the previous iteration. Of course, if the pickup of a request has been already visited or a courier is currently moving towards it, then this request cannot be reinserted into another route.

Table 4.1. Feasible route schedules associated to customer sequencing $(i_1^-, i_3^+, i_2^-, i_3^-)$.

| Total waiting time along the route | Waiting time at $i_1^-$ | Waiting time at $i_3^+$ | Waiting time at $i_2^-$ | Waiting time at $i_3^-$ | Home positions |
|---|---|---|---|---|---|
| 0 | - | - | - | - | $i_q, i_{q'}$ |
| 10 | 10 | - | - | - | $i_q, i_{q'}$ |
| | - | 10 | - | - | |
| | - | - | 10 | - | |
| | - | - | - | 10 | |
| 20 | 20 | - | - | - | $i_q, i_{q'}$ |
| | - | 20 | - | - | |
| | - | - | 20 | - | |
| | - | - | - | 20 | |
| | 10 | 10 | - | - | |
| | 10 | - | 10 | - | |
| | 10 | - | - | 10 | |
| | - | 10 | 10 | - | |
| | - | 10 | - | 10 | |
| | - | - | 10 | 10 | |

## 4.4 Experimental Results

Our computational experiments aim at assessing the effectiveness of our anticipatory algorithms versus a purely reactive procedure. Both heuristics have been coded in C++ and run on a PC with a Pentium IV processor clocked at 2.8 GHz. We have utilized two sets of randomly generated instances, resembling standard and premium service, respectively. In the former instances ("S" instances, in the following), a courier may consolidate an unlimited number of deliveries. In the latter instances ("P" instances, in the following), consolidation is not allowed, so that vehicles may be assumed to have capacity equal to one. In both cases, the service territory is modelled as a grid with 25 zones, giving rise to 600 origin-destination pairs. Travel times between adjacent zones have been set equal to 15 minutes. Demands are spatially independent and uniformly distributed over $Q = \{(o,d) \in \{1,\dots,25\}^2 : o \neq d\}$. For each origin-destination pair, their arrival times constitute a Poisson process with rate $\lambda_{o,d}$ over a planning horizon [0,480] minutes. The arrival rates $\lambda_{o,d}$ are set equal to a constant $c$ for 20 randomly chosen origin-destination pairs $(o,d) \in Q_1 \subseteq Q$ and to $c/k$ for the remaining 580 pairs, where $c$ and $k$ are determined in such a way that the expected overall number of requests $\mu$ is 200 and the expected number of requests $\mu_1$ between pairs in $Q_1$ is given the following values: 0, 50, 100, 150, 200. Moreover, we assume that the inconvenience associated to each request is represented by its waiting time. In both sets, the number of vehicles is equal to 35. For each set and for each value of $\mu_1$, we generated as many instances as are required to keep the variation coefficient of the sample mean $(\sigma/\sqrt{n})/|\overline{x}|$ of the objective function less than 0.1, where $\sigma$ denotes the sample standard deviation, $n$ the number of instances and $\overline{x}$ the sample mean. Moreover, algorithm parameters have been set as follows: $\kappa = 5$, $\delta = 20$ minutes, $\alpha = 0.1$, $n_0 = 10$, $MWT = 20$ minutes, $WQ = 10$ minutes. Finally, we allow idle vehicles to be relocated to any of the three stochastic medians of the graph ($|H| = 3$).

The first aim of our computational experiments is to correlate empirically the duration $\Delta t_k$ of the short-term horizon to the problem data. Intuitively, as $\Delta t_k$ increases, we expect that the procedure becomes less and less myopic while the computational effort turns out to be heavier. On the other hand, very large $\Delta t_k$ values do not provide any relevant additional objective function improvement. Tables 4.2 and 4.3, as well as Figures 4.2 and 4.3, provide an account of the results whereas $\mu_1 = 200$, in which case the expected service time is around 40 minutes. The column headings are as follows:

- $n$: number of instances generated;
- $z_R$: average objective function value for the reactive procedure ($\frac{\sum_{i=1}^{n} z_{R,i}}{n}$, where $z_{R,i}$ is the objective function value provided by the reactive procedure for instance $i$);
- $z_A$: average objective function value for the anticipatory insertion procedure ($\frac{\sum_{i=1}^{n} z_{A,i}}{n}$, where $z_{A,i}$ is the objective function value provided by the anticipatory insertion procedure for instance $i$);

- $TIME_R$: average CPU time (in seconds) for the reactive procedure;
- $TIME_A$: average CPU time (in seconds) for the anticipatory insertion procedure;
- $SERVTIME_R$: average CPU time per request (in seconds) for the reactive procedure;
- $SERVTIME_A$: average CPU time per request (in seconds) for the anticipatory insertion procedure;
- $\rho_R$: average utilization rate of a vehicle for the reactive procedure;
- $\rho_A$: average utilization rate of a vehicle for the anticipatory insertion procedure;
- $OBJDEV$: average deviation of the objective function values for the reactive and anticipatory insertion procedures, i.e. $\frac{z_A - z_R}{z_R}$;
- $OBJSSD$: sample standard deviation of the objective function values for the reactive and anticipatory insertion procedures, i.e. $\sqrt{\frac{\sum_{i=1}^{n}(z_{R,i} - z_{A,i})^2}{n-1}}$;
- $CIW$: width of the 0.9 confidence interval for the difference of the objective function values provided by the reactive and anticipatory insertion procedures.

These results, along with more extensive computational experiments on both "S" and "P" instances (which we do not report for the sake of brevity), show that a good trade-off between computation time and solution quality can be achieved for $\Delta t_k$ values close to the expected service time at $t_k$.

The second part of our experimental study aims at comparing the anticipatory insertion algorithm with its reactive counterpart. Tables 4.4 and 4.5 show the results for the "S" and "P" instances, respectively. The column heading which has not been explained yet is as follows:

- $SAMPLES$: average number of samples generated for each instance.

The results show that the anticipatory insertion procedure provides consistently better solutions than a purely reactive procedure on all generated instances. More

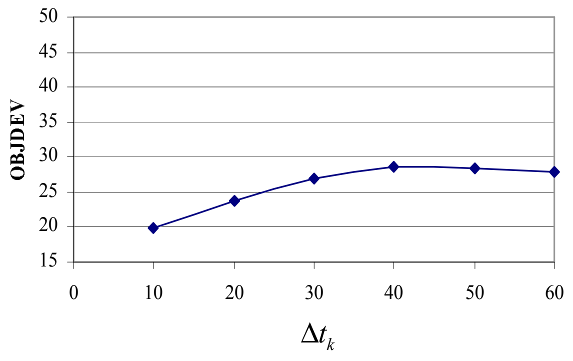


Figure 4.2. Average deviation of the objective function values for the reactive and anticipatory procedures as a function of $\Delta t_k$ (“S” instances).

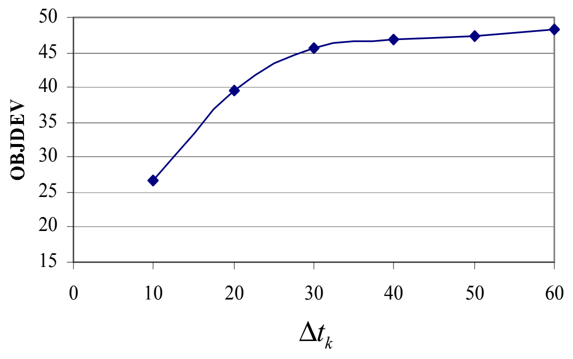


Figure 4.3. Average deviation of the objective function values for the reactive and anticipatory procedures as a function of $\Delta t_k$ (“P” instances).

precisely, the anticipatory algorithm allows achieving an average improvement of the objective function value ranging between 11% and 25% for the “S” instances, whereas for the “P” instances our procedure reduces the objective function value by 38% to 59%. It is worth noting that the use of the anticipatory algorithm leads to an average utilization rate which is around 19% lower than its reactive counterpart. In conclusion, the experiments show that the anticipatory procedure is very effective, leading to a reduction in the customer inconvenience and to an evener distribution of the requests among the vehicles.

In the third part of our experimental study, we have compared the anticipatory insertion and local search algorithms. For the sake of brevity, we report only the results on the “S” instances. (Table 4.6). The column headings which have not been explained yet are as follows:

- $z_L$: average objective function value for the anticipatory local search procedure ($\frac{\sum_{i=1}^{n} z_{L,i}}{n}$, where $z_{L,i}$ is the objective function value provided by the anticipatory local search procedure for instance $i$);
- $TIME_L$: average CPU time (in seconds) for the anticipatory local search procedure;
- $SERVTIME_L$: average CPU time per request (in seconds) for the anticipatory local search procedure;
- $\rho_L$: average utilization rate of a vehicle for the anticipatory local search procedure;

The experiments show that in a typical same-day courier setting, the anticipatory local search allows to yield only very slight improvements (if any) on the simpler anticipatory insertion policy. This behavior can be explained as follows: since nowadays the service level provided by same-day couriers is quite high, at any time each vehicle is usually assigned a small number of pending requests which limits the number of feasible solutions.

## 4.5 Conclusions

The dynamic *Vehicle Dispatching Problem with Pickups and Deliveries* is faced by local area courier companies serving same-day pickup and delivery requests for the transport of letters and small parcels. For this problem we have proposed an anticipatory mechanism which we have then embedded in both an insertion and a local search algorithm. A major challenge we had to face when solving large instances was to keep the computational effort low. This was achieved through two main features: firstly, the number of demand samples was determined through an IZS procedure that allows to eliminate, at an early stage of experimentation, those solutions that are clearly inferior; secondly, we limited the computational effort by applying the sampling procedure to a short-term horizon whose duration was empirically correlated to the problem data. Another advantage of our approach is that it addresses in an unified and integrated way all the main issues involved in real-time fleet management (a similar approach was proposed by Bent and Van Hentenryck (2007) almost simultaneously to our thesis). Computational results have shown that the anticipatory insertion algorithm provides consistently better solutions than its reactive counterpart. They have also proved that, in a typical same-day courier setting, an anticipatory local search procedure allows to yield only very slight improvements on a simpler anticipatory insertion policy.

In this chapter we have assumed that the requests arrive according to a known stochastic process. Future work should be aimed at verifying how robust our approach is whereas demand sampling is based on an approximation of the arrival process.

Table 4.2. Preliminary computational experiments on “S” instances.

| $\Delta t$ | $n$ | $z_R$ | $z_A$ | $TIME_R$ | $TIME_A$ | $SERV$ $TIME_R$ | $SERV$ $TIME_A$ | $\rho_R$ | $\rho_A$ | $OBJ$ $DEV$ | $OBJ$ $SSD$ | $\frac{\sigma/\sqrt{n}}{\lvert\overline{x}\rvert}$ | $CIW$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | 4 | 1692 | 1309 | 1.2 | 5454 | 0.01 | 26.1 | 0.63 | 0.35 | -0.22 | 0.03 | 0.07 | 0.06 |
| 20 | 5 | 1729 | 1318 | 1.8 | 6891 | 0.01 | 33.5 | 0.63 | 0.35 | -0.23 | 0.04 | 0.08 | 0.07 |
| 30 | 5 | 1729 | 1265 | 1.8 | 8120 | 0.01 | 39.4 | 0.63 | 0.36 | -0.26 | 0.04 | 0.06 | 0.07 |
| 40 | 5 | 1729 | 1234 | 1.8 | 9378 | 0.01 | 45.5 | 0.63 | 0.36 | -0.28 | 0.03 | 0.04 | 0.04 |
| 50 | 5 | 1729 | 1237 | 1.8 | 10725 | 0.01 | 52.1 | 0.63 | 0.36 | -0.28 | 0.02 | 0.04 | 0.04 |
| 60 | 5 | 1729 | 1244 | 1.8 | 11957 | 0.01 | 58.1 | 0.63 | 0.37 | -0.28 | 0.02 | 0.04 | 0.05 |
|  |  |  |  |  |  |  |  |  |  | -0.26 |  |  |  |

Table 4.3. Preliminary computational experiments on “P” instances.

| $\Delta t$ | $n$ | $z_R$ | $z_A$ | $TIME_R$ | $TIME_A$ | $SERV$ $TIME_R$ | $SERV$ $TIME_A$ | $\rho_R$ | $\rho_A$ | $OBJ$ $DEV$ | $OBJ$ $SSD$ | $\frac{\sigma/\sqrt{n}}{\lvert\overline{x}\rvert}$ | $CIW$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | 4 | 1736 | 1271 | 1.0 | 806 | 0.001 | 3.7 | 0.63 | 0.49 | -0.26 | 0.02 | 0.05 | 0.05 |
| 20 | 5 | 1802 | 1078 | 1.2 | 2527 | 0.1 | 12.1 | 0.63 | 0.46 | -0.39 | 0.07 | 0.09 | 0.13 |
| 30 | 5 | 1802 | 967 | 1.2 | 1942 | 0.1 | 9.3 | 0.63 | 0.45 | -0.45 | 0.08 | 0.09 | 0.15 |
| 40 | 5 | 1802 | 947 | 1.2 | 2597 | 0.1 | 12.5 | 0.63 | 0.46 | -0.47 | 0.07 | 0.07 | 0.14 |
| 50 | 4 | 1736 | 912 | 1.0 | 3259 | 0.001 | 15.7 | 0.63 | 0.46 | -0.47 | 0.04 | 0.05 | 0.09 |
| 60 | 5 | 1802 | 925 | 1.2 | 3973 | 0.1 | 19.1 | 0.63 | 0.46 | -0.48 | 0.06 | 0.06 | 0.11 |
|  |  |  |  |  |  |  |  |  |  | -0.42 |  |  |  |

Table 4.4. Comparison between the anticipatory and reactive insertion algorithms (“S” instances).

| $\mu_1$ | $n$ | $SAMPLES$ | $z_R$ | $z_A$ | $TIME_R$ | $TIME_A$ | $SERV$ $TIME_R$ | $SERV$ $TIME_A$ | $\rho_R$ | $\rho_A$ | $OBJ$ $DEV$ | $OBJ$ $SSD$ | $\frac{\sigma/\sqrt{n}}{\lvert\overline{x}\rvert}$ | $CIW$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 200 | 5 | 12.4 | 1729 | 1302 | 2.1 | 4743 | 0.01 | 23.04 | 0.63 | 0.37 | -0.25 | 0.03 | 0.06 | 0.026 |
| 150 | 5 | 11.9 | 1751 | 1454 | 1.8 | 4738 | 0.01 | 22.75 | 0.62 | 0.39 | -0.17 | 0.04 | 0.10 | 0.035 |
| 100 | 5 | 13.1 | 1770 | 1511 | 1.8 | 4683 | 0.01 | 23.26 | 0.62 | 0.45 | -0.15 | 0.05 | 0.10 | 0.047 |
| 50 | 5 | 14.2 | 1795 | 1599 | 1.4 | 4913 | 0.01 | 24.79 | 0.60 | 0.46 | -0.11 | 0.06 | 0.10 | 0.051 |
| 0 | 4 | 13.4 | 1814 | 1601 | 1.3 | 5109 | 0.01 | 26.04 | 0.58 | 0.44 | -0.18 | 0.02 | 0.10 | 0.021 |
| | | | | | | | | | | | -0.17 | | | |

Table 4.5. Comparison between the anticipatory and reactive insertion algorithms (“P” instances).

| $\mu_1$ | $n$ | $SAMPLES$ | $z_R$ | $z_A$ | $TIME_R$ | $TIME_A$ | $SERV$ $TIME_R$ | $SERV$ $TIME_A$ | $\rho_R$ | $\rho_A$ | $OBJ$ $DEV$ | $OBJ$ $SSD$ | $\frac{\sigma/\sqrt{n}}{\lvert\overline{x}\rvert}$ | $CIW$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 200 | 5 | 13.6 | 1667 | 936 | 1.7 | 2563 | 0.01 | 12.35 | 0.64 | 0.45 | -0.44 | 0.05 | 0.08 | 0.056 |
| 150 | 5 | 12.8 | 1691 | 830 | 2.0 | 2350 | 0.01 | 11.34 | 0.63 | 0.46 | -0.50 | 0.11 | 0.10 | 0.104 |
| 100 | 4 | 11.7 | 1366 | 707 | 1.8 | 2214 | 0.01 | 11.11 | 0.63 | 0.43 | -0.59 | 0.09 | 0.10 | 0.085 |
| 50 | 5 | 13.5 | 1274 | 787 | 1.8 | 2090 | 0.01 | 10.69 | 0.63 | 0.45 | -0.38 | 0.03 | 0.09 | 0.029 |
| 0 | 4 | 14.1 | 1391 | 726 | 2.2 | 2156 | 0.01 | 11.11 | 0.65 | 0.46 | -0.48 | 0.08 | 0.10 | 0.083 |
| | | | | | | | | | | | -0.48 | | | |

Table 4.6. Comparison between the anticipatory insertion and local search algorithms (“S” instances).

| $\mu_1$ | $n$ | *SAMPLES* | $z_A$ | $z_L$ | $TIME_A$ | $TIME_L$ | *SERV* $TIME_A$ | *SERV* $TIME_L$ | $\rho_A$ | $\rho_L$ | *OBJ DEV* | *OBJ SSD* | $\frac{\sigma/\sqrt{n}}{\lvert\overline{x}\rvert}$ | *CIW* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 200 | 5 | 12.8 | 1302 | 1258 | 4743 | 5167.6 | 23.04 | 25.11 | 0.37 | 0.33 | -0.03 | 0.02 | 0.10 | 0.017 |
| 150 | 5 | 12.2 | 1454 | 1445 | 4738 | 5054.2 | 22.75 | 24.23 | 0.39 | 0.38 | -0.01 | 0.01 | 0.08 | 0.001 |
| 100 | 5 | 13.4 | 1511 | 1502 | 4683 | 5296.6 | 23.26 | 26.34 | 0.45 | 0.43 | -0.01 | 0.03 | 0.10 | 0.030 |
| 50 | 5 | 14.1 | 1599 | 1519 | 4913 | 5063.8 | 24.79 | 25.56 | 0.46 | 0.41 | -0.05 | 0.06 | 0.09 | 0.057 |
| 0 | 4 | 13.7 | 1601 | 1592 | 5109 | 5124.0 | 26.04 | 26.09 | 0.44 | 0.43 | -0.01 | 0.03 | 0.09 | 0.029 |
|  |  |  |  |  |  |  |  |  |  |  | -0.02 |  |  |  |

# Chapter 5

# The Shift Scheduling Problem in the Same-day Courier Industry

## 5.1 Introduction

The courier industry is an integral part of the industrialized countries' economy, providing transportation of documents, packages, machine parts, medical supplies and other time-sensitive goods among companies and other organizations. Numerous businesses rely heavily on couriers: financial institutions that must transfer documents between branches and processing centers, law firms that need to deliver confidential documents on very strict deadlines, pharmaceutical distributors willing to transport high valued medications to hospitals, etc.

The courier market can be divided into two components: companies with their main focus being national and international overnight deliveries, and local messenger companies, whose main focus is to provide same-day deliveries in smaller well-defined geographical areas (e.g., a large metropolitan area or a state). A few figures can be used to illustrate the economic relevance of this sector. In the US there are approximately seven thousand courier companies that generate a multi-billion dollar revenue (Messenger Courier Association of the Americas, 2006). In Canada the courier industry is estimated to be worth 5.4 billion dollars in sales, and is expected to grow at 2.6 per cent in volume and 5.9 per cent in revenue during next few years (Statistics Canada, 2004, Canadian Courier & Messenger Association, 2004). Same-day courier companies generate 15-20 per cent of the industry revenue in most industrialized countries (see, e.g., Transport Canada, 2003). While the national and international overnight market is dominated by a few multinational corporations (including DHL, UPS, FedEx and USPS) and by national-wide companies (like Purolator in Canada and Bartolini in Italy), the local messenger sector consists almost entirely of small, locally owned and operated businesses. Moreover, it is worth noting that the former market is organized rationally and efficiently (couriers usually make between 15 and 25 deliveries per hour), whereas in the latter market couriers may only make 15-20 deliveries *in a day* that are often spread out over a large urban area.

Same-day clients usually request couriers with little or no notice, all but eliminating the ability to construct routes or schedules in advance. Once a courier has been assigned a job, he/she proceeds directly to the pickup location, collects the appropriate conveyance, and moves on to the delivery where in return, a signature is obtained. The traditional model of same-day courier service utilizes human controllers who communicate with bicycle, motorcycle, car and van couriers via radio or mobile phone. Controllers ask couriers to relay their location information and then assign jobs to the closest and most appropriate courier. This model is not only inefficient, but also suffers from errors inherent with the involvement of a human element. As the number of couriers increases to several hundreds or thousands, informational complexity grows to levels which push the limits of human analysis. At this stage several controllers may be used, increasing costs and requiring constant coordination between controllers in addition to communication with couriers on the street. Human controllers can generally manage a maximum of thirty couriers in a specific area of the city and make job allocation decisions based on incomplete or inaccurate information. This has contributed to the fact that no courier company can gain a very large market share, due to the informational complexity of the allocation problem. For instance, in London (UK) the industry (which is worth about 450 million pounds) is split among about 600 suppliers (Marketing Research for Industry, 2006).

Automated information-based job allocation systems (Attanasio et al., 2007) are based on dispatching algorithms able to assign each job to the most appropriate courier on the basis of the current fleet location and status. Courier location information is provided by GPS devices (Cathey and Dailey, 2003) embedded into palmtop computers which are also used to provide directions to couriers. Advantages of such systems include an improvement of the courier efficiency, a reduction of the requirements of human supervisors as well as the possibility to provide customers some sort of *quality of service* (QoS) guarantee.

In this chapter we deal with the same-day *Courier Shift Scheduling Problem* (CSSP), a tactical problem which amounts to minimize the staffing cost subject to probabilistic service level requirements. In what follows we assume that couriers are independent contractors paid by the hour, thus creating economic incentives for companies to hire the least amount of labour possible. Since the demand is random in nature, this problem is intrinsically different from crew scheduling problems encountered in other transportation areas such as the railway (Caprara et al., 2006) and airline (Gopalakrishnan and Johnson, 2005) industries. Moreover, the CSSP differs from other shift scheduling problems (like those found in the call center industry, Gans et al., 2003) for several reasons, including the duration of jobs which tipically spans several hours. As explained subsequently, this feature of the CSSP forbids to exploit most results obtained so far in other settings.

## 5.2 Problem Formulation

At an operational level, same day couriers have to solve a *Dynamic Vehicle Dispatching Problem with Pickups and Deliveries* (Gendreau et al., 2006, Ghiani et al., 2007) which amounts to allocate each request to a vehicle as well as to schedule the requests assigned to each vehicle route. A pickup and its associated delivery must be satisfied by the same vehicle and a pickup must always be made before its associated delivery. The objective is to minimize the total expected customer inconvenience. For this problem, Gendreau et al. (2006) describe a reactive Tabu Search procedure while Ghiani et al. (2007) present an anticipatory mechanism based on near-future simulation and ranking & selection procedures.

Given any dispatching policy $P$, at a tactical level courier companies have to decide how many couriers should be allocated to each shift pattern subject to QoS constraints (*Courier Shift Scheduling Problem*, CSSP). The CSSP is usually solved on a weekly or quarterly basis (the demand is usually characterized by significant yearly/weekly/daily seasonal effects) with the aim to minimize the staffing cost. In what follows, we assume that a QoS guarantee is provided to those requests arriving into a planning horizon $H$ (e.g., Monday to Friday, 6.00 am to 10.00 pm) and that the QoS is assessed with respect to a subset $J$ of time intervals (*QoS intervals*) in which $H$ is partitioned (Figure 5.1).

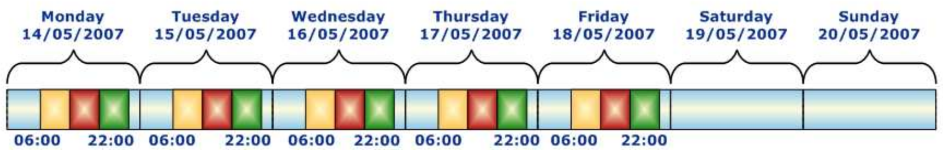


Figure 5.1. A sample planning horizon $H$ and its partitioning into 15 QoS intervals.

In particular, we require that the expected service time of a request arriving during time interval $j$ ($j = 1,\dots,|J|$) must be less than a given threshold $T_j$. In a real-world setting, thresholds may also depend on the customer class (e.g., they can be tighter for customers having stipulated a *Service Level Agreement* (SLA) and looser for spot customers). In addition, they may depend on both the pickup and delivery locations (e.g., tighter thresholds may be assigned to shipments inside a central urban zone). In the remainder of the chapter, we neglect these dependencies for the sake of simplicity, although the subsequent algorithms can be easily adapted *mutatis mutandis* to the more general case.

In this chapter we assume that the feasible shifts (i.e., shift satisfying rest regulations) can be enumerated. This assumption is quite realistic since only few patterns are acceptable in real world (e.g., shifts covering 4 days a week for 10 hours per day *or* 5 days a week for 8 hours per day (Attanasio et al., 2007). Let $Q$ be the set of feasible shift patterns and let $c_q$ be the wage of a courier covering pattern $q$ ($q = 1,\dots,|Q|$). The CSSP amounts to determine the optimal number $x_q$ of drivers covering shift pattern $q$ ($q = 1,\dots,|Q|$):

$$\text{(5.1)} \qquad \text{Minimize} \qquad \textstyle\sum_{q=1}^{|Q|} c_q x_q$$

$$\text{(5.2)} \qquad \text{s.t.} \qquad g_j(x_1,\dots,x_Q,\xi) \leq T_j \qquad j = 1,\dots,|J|$$

$$\text{(5.3)} \qquad x_q \geq 0, \ \text{integer} \qquad q = 1,\dots,|Q|$$

where $\xi$ is a vector denoting the random demands across the planning horizon and $g_k(x,\xi)$ is the expected *service* (or *system*) *time* of a request arising during QoS interval $j$ ($j = 1,\ldots,|J|$). The complexity of the model lies in the $g_j(\cdot,\cdot)$ functions, which are non linear and not known explicitly.

Formulation (5.1)-(5.3) is similar to models employed in call center (CC) shift scheduling modelling (Gans et al., 2003), except that in our application the QoS is measured as an expected service time while in call center applications it is assessed by means of expected number of reneging customers (no customer usually drops in a courier application). Apart for this minor issue, the two shift scheduling problems differ for the order of magnitude of service times, which is approximately hundreds of minutes in the courier industry and around a few minutes in the call center sector. This characteristic has a profound impact on the solution strategies. Indeed, except for Avramidis et al. (2007), the call center shift problem is decomposed into a *staffing problem*, which amounts to determine the number of employees needed at any time (e.g., between 10 am and 11 am) and a *set covering problem* (Caprara et al., 2000). The staffing problem is solved, depending on the number of customer classes and the specialization of agents, by reversing steady state queuing formulae or through simulation-based cutting plane algorithms. See Avramidis and L'Ecuyer (2005), Atlason et al. (2004a,b) for recent contributions in this area of research.

## 5.3 Approximate Neighborhood Evaluation Procedure

Because of the magnitude of service times, the CSSP cannot be decomposed into simpler subproblems as its call center counterpart. This observation has motivated us to deal with the CSSP as a whole. The main computational difficulty when devising a neighborhood search procedure for the CSSP is the peculiar nature of the QoS functions $g_j(\cdot,\cdot)$ that are not known explicitly and can only be estimated through simulation for any given vector $x = (x_1,\ldots,x_Q)$. Since simulating courier operations is very time consuming even for simple dispatching policies $P$ (Ghiani et al., 2007), our effort has been devoted to explore the search space efficiently. Let $x^{(k)}$ be the current solution at iteration $k$ of a neighborhood search procedure and let $N(x^{(k)})$ be its neighborhood. In principle, we could select $x^{(k+1)}$ as the least cost feasible solution in $N(x^{(k)})$. This approach could be implemented by checking the feasibility (e.g., the satisfaction of QoS constraints (5.2) through simulation) of the least cost solution. If this check succeeds, we are done. Otherwise, we should check the second least cost solutions, etc. Since the best solutions in the neighborhood are likely to be infeasible, this approach might result in the examination of a large number of solutions, thus requiring many time consuming simulation experiments. On the other hand, a procedure picking up $x^{(k+1)}$ at random in $N(x^{(k)})$ would perform poorly in practice, as shown in Section 5.4. Trading off between these two extremes, we propose a procedure that collects some statistics when simulating $x^{(k)}$ at iteration $(k-1)$. Then we use these statistics into an *Approximated Neighborhood Evaluation* (ANE) procedure that approximates the QoS functions $g_j(\cdot,\cdot)$ with deterministic linear functions of the $x$ variables. We divide the planning horizon into $m$ micro-intervals $I_h$ of duration $\mathit{\Delta} t$ (for instance, $\mathit{\Delta} t = 30$ minutes or 1 hour) and assume that the arrival rate $\lambda_h$ during micro-interval $I_h$ $(h = 1,\dots,m)$ is constant. Let $a$ and $d$ be the arrival time and the delivery time of a request. When selecting $x^{(k)}$ as the new current solution at iteration $(k-1)$, we compute an estimation $\hat{s}_{hl}$ of the conditional expected value of the service time of the requests arriving in micro-interval $I_h$ and serviced in $I_l$ $(l \geq h)$:

$$s_{hl} = E[s|a \in I_h, d \in I_l, x^{(k)}].$$

It is worth noting that such an estimate comes at no cost since the QoS provided by $x^{(k)}$ has to be evaluated (through simulations runs) before this solution can be declared feasible.

Let $H_j$ be the set of micro-intervals which made up QoS interval $j$ $(j = 1,\dots,|J|)$. By using the well known total probability theorem, constraints (5.2) can be reformulated as follows:

$$\sum_{h \in H_j} \left( \frac{\lambda_h}{\sum_{i \in H_j} \lambda_i} \sum_{l=h,h+1,\dots} \frac{\lambda_{hl}}{\lambda_h} \hat{s}_{hl} \right) \leq T_j, \qquad j = 1,\dots,|J|$$

where $\lambda_{hl}$ is the part of $\lambda_h$ delivered during micro-interval $l$ $(l \geq h)$ for any neighbor $x \in N(x^{(k)})$. It is worth noting that we are assuming implicitly that conditional expected values $s_{hl}$ do not vary significantly in $N(x^{(k)})$, which is quite reasonable if the size $|N(x^{(k)})|$ of the neighborhood is "small enough". If no data have been collected for some $h-l$ pairs, we can assume uniformity. For example, if the duration of micro-intervals is one hour and time is measured in hours, we can set:

$$\hat{s}_{hl} = 0.5 + h - l$$

We can now formulate the ANE model:

$$
\begin{array}{lll}
(5.4) & \text{Minimize} & \sum_{q=1}^{|Q|} c_q x_q \\
(5.5) & \text{s.t.} & \sum_{h\in H_j}\left(\frac{1}{\sum_{i\in H_j}\lambda_i}\sum_{l=h,h+1,\dots}\lambda_{hl}\hat{s}_{hl}\right) \le T_j, \quad j=1,\dots,|J| \\
(5.6) & & \sum_{l=h,h+1,\dots}\lambda_{hl} = \lambda_h, \quad h=1,\dots,m \\
(5.7) & & \sum_{h=1}^{l}\lambda_{hl} \le \eta_l \sum_{q\in A_l} x_q, \quad l=1,\dots,m \\
(5.8) & & d(x,x^{(k)}) \le |N(x^{(k)}| \\
(5.9) & & \lambda_{hl} \ge 0, \quad h=1,\dots,m; \quad l=h,h+1,\dots,m \\
(5.10) & & x_q \ge 0, \ \text{integer}, \quad q=1,\dots,|Q|
\end{array}
$$

where the objective function (5.4) is the same as in formulation (5.1)-(5.3), (5.5) are the previously introduced QoS linearly approximated constraints, (5.6) are the flow conservation constraints, (5.7) are capacity constraints, while (5.8) imposes that the distance $d(x,x^{(k)})$ between $x$ and the current solution $x^{(k)}$ does not exceed the neighborhood size $|N(x^{(k)})|$. In particular, constraints (5.7) state that the expected demand serviced in $I_l$ does not exceed the number of couriers working in $I_l$ multiplied by courier productivity $\eta_l$. Productivity may vary significantly during the day since higher demand rates usually give couriers the opportunity to consolidate more deliveries without deteriorating the QoS level (see Genta and Muñoz (2007) for an account of some related issues). Parameters $\eta_l$ are not well defined because: (a) vehicles may transport more that a single load at a time; (b) a request may span two or more micro-intervals. In subsections 5.3.1 and 5.3.2 we describe two methods for estimating these parameters: the first approach is based on an intuitive load assignment procedure while the second solves a fitting problem in $x^{(k)}$. We define the distance $d(x,x^{(k)})$ between $x$ and the current solution $x^{(k)}$ in a quite natural way as

$$d(x,x_q^{(k)}) = \sum_{q=1}^{|Q|} |x_q - x_q^{(k)}|$$

Hence constraints (5.8) can be casted in a linear form by expressing each $x_q$ variable as the sum of $x_q^{(k)}$ and a linear combination of a set of binary variables $y_{rq}$ and $w_{rq}$, weighted by both positive and negative powers of 2:

$$x_q - x_q^{(k)} + \sum_{r=0,1,\ldots} 2^r y_{rq} - \sum_{r=0,1,\ldots} 2^r w_{rq}, \quad q = 1,\ldots,|Q| \tag{5.11}$$

$$y_{rq}, x_{rq} \in (0,1), \quad q = 1,\ldots,|Q|; r = 0,1,\ldots \tag{5.12}$$

Thus $x_q$ nonnegativity constraints become

$$\sum_{r=0,1,\ldots} 2^r w_{rq} \leq x_q^{(k)} \qquad q = 1,\ldots,|Q| \tag{5.13}$$

while (5.8) are casted as

$$\sum_{q\in Q} \sum_{r=0,1,\ldots} 2^r y_{rq} + \sum_{q\in Q} \sum_{r=0,1,\ldots} 2^r w_{rq} \leq |N(x^{(k)}| \tag{5.14}$$

The ANE formulation (5.4)-(5.7), (5.9)-(5.14) is then solved by means of an off-the-shelf general purpose Integer Linear Programming solver as in the spirit of Fischetti and Lodi (2004).

### 5.3.1 Naïve load assignment procedure

The first approach for estimating productivity $\eta_l$ is based on an intuitive procedure in which, for each sample taken at $x^{(k)}$, it is computed the contribution given by micro-interval $I_l$ to the service of each request. The contributions associated to the various requests are then summed up and are averaged over all samples taken at $x^{(k)}$. In this procedure the computation of the productivity in micro-interval $I_l$ for a single sample is:

$$\eta_l = \sum_j \eta_{lj}$$

where $j$ are all the requests whose service starts or is completed within micro-interval $I_l$. The quantities $\eta_{lj}$ are computed as follows:

$$\eta_{lj} = \frac{\text{amount of time of } I_l \text{ spent in serving request } j}{\text{whole time needed to serve } j}$$

For the sake of clarity, in Figures 5.2 and 5.3 we report two typical situations. More specifically, Figure 5.2 describes a situation in which a single request $k$ is present during all the micro-interval $I_l$. In this case the computation of $\eta_l$ would be:

$$\eta_l = \eta_{lk} = \frac{60}{40} = 1.5[\text{requests/h}]$$

It is not surprising that, although $I_l$ contains a single request, the productivity is greater than 1. The vehicle is idle for a certain amount of time, which allows it to potentially serve other requests.

Figure 5.3, instead, reports a slightly more complicated route, in which three requests, namely $s$, $k$ and $j$, whose service starts or ends during micro-interval $I_l$. Thus, $\eta_l$ is computed as:

$$\eta_l = \eta_{ls} + \eta_{lk} + \eta_{lj} = \frac{20'}{40' + 20'} + \frac{50'}{50' + 20'} + \frac{40'}{40' + 30'} = 1.61[\text{requests/h}]$$

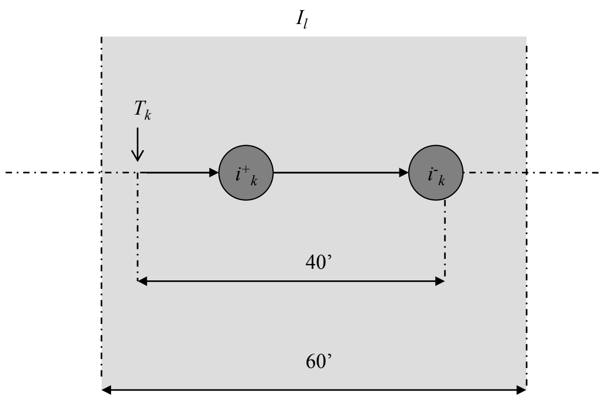


Figure 5.2. A sample route in which a single request is served during $I_l$.

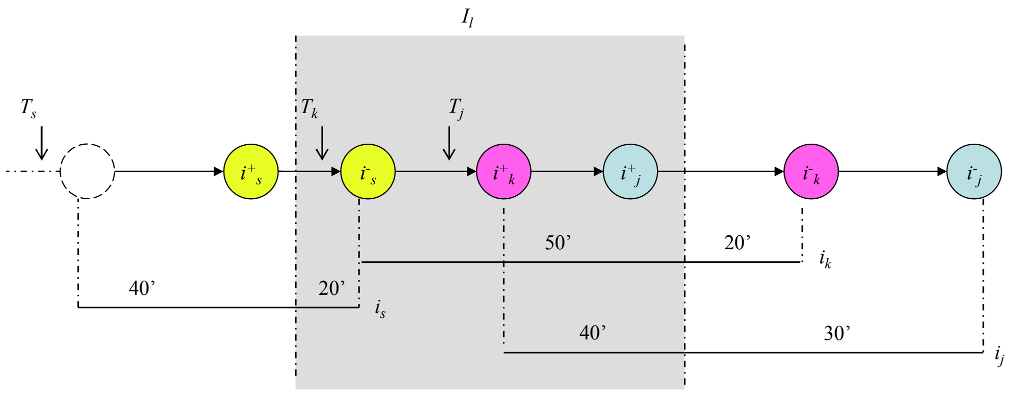


Figure 5.3. A sample route in which more requests are served during $I_l$.

## 5.3.2 Fitting Procedure

As observed, the choices made by the load assignment procedure described in the previous subsection are quite arbitrary and this may impact on the solution of the ANE approach. Hence, we have devised an alternative approach which fits the ANE model in $x^{(k)}$. Let $\hat{\lambda}_{hl}$ be the estimated flows in $x^{(k)}$ and let $\lambda_{hl}$ be their values predicted by the ANE approach. We determine the values of the $\eta_l$ parameters in such a way that the absolute error $\sum_{h=1,2,\dots}\sum_{l=h,h+1,\dots}|\lambda_{hl}-\hat{\lambda}_{hl}|$ of the ANE approach in $x^{(k)}$ is minimized

$$\begin{array}{lll}
(5.15) & \text{Minimize} & \sum_{h=1,2,\dots}\sum_{l=h,h+1,\dots}|\lambda_{hl}-\hat{\lambda}_{hl}| \\
(5.16) & \text{s.t.} & \sum_{l=h,h+1,\dots}\frac{\lambda_{hl}}{\lambda_h}\hat{s}_{hl}=\hat{s}_h, \quad h=1,2\dots \\
(5.17) & & \sum_{l=h,h+1,\dots}\lambda_{hl}=\lambda_h, \quad h=1,2\dots \\
(5.18) & & \sum_{h=1}^{l}\lambda_{hl}\le\eta_l\sum_{q\in A_l}x_q, \quad l=1,2\dots, \\
(5.19) & & \lambda_{hl}\ge 0, \quad h=1,\dots; \quad l=h,h+1,\dots \\
(5.20) & & \eta_l\ge 0 \quad l=1,2\dots
\end{array}$$

In this problem, constraints (5.16) and (5.17) impose that predicted flow in $x^{(k)}$ satisfy the total probability theorem (wrt expected service times) and flow conservation. Constraints (5.18) are obtained by imposing (5.7) in $x^{(k)}$ while considering $\eta_l$ as non-negative variables. It is easy to cast problem (5.15)-(5.20) as a linear program by using a simple transformation of the objective function (see, e.g., Hillier and Lieberman, 2004). Moreover, it is worth noting that the fitting problem (5.15)-(5.20) is always feasible. Indeed, for $\eta_l\to\infty$ $(l=1,2,\dots)$, the remaining constraints express each expected value $\hat{s}_h$ as a convex combination of conditional expected values $\hat{s}_{hl}$ $(l=h,h+1,\dots)$:

$$\sum_{l=h,h+1,\dots}\frac{\lambda_{hl}}{\lambda_h}\hat{s}_{hl}=\hat{s}_h, \qquad \sum_{l=h,h+1,\dots}\frac{\lambda_{hl}}{\lambda_h}=1, \quad \frac{\lambda_{hl}}{\lambda_h}\ge 0.$$

These constraints are always satisfied since $\hat{s}_{hh}\le\hat{s}_{hh+1}\le\hat{s}_{hh+2}\le\dots$ for every $h=1,2,\dots$.

## 5.4 Computational Results

In this section we describe the computational experiments performed in order to assess the effectiveness of the ANE mechanism.

Because the problem in this chapter has not been previously addressed in the literature with reference to the courier industry, data set generation is required. For this purpose we consider a number of randomly generated instances, resembling standard courier service, in which a courier may consolidate an unlimited number of deliveries. The service territory is modelled as a grid with 25 zones, giving rise to 600 origin-destination pairs. Travel times between adjacent zones are set equal to 15 minutes. The planning horizon $H$ is made up of five days, from Monday to Friday, 8.00 am to 8.00 pm. Each day is partitioned into 3 QoS intervals (8.00 am to 12.00 pm, 12.00 pm to 4.00 pm, 4.00 pm to 8.00 pm), so that the overall number of QoS intervals is 15 ($|J| = 15$). For each QoS interval $j$, the threshold for the expected service time, exceeded which the solution is considered infeasible, is constant and equal to 120 minutes. Moreover each QoS interval is divided into 4 micro-intervals made up of 60 minutes, in such a way that the overall number of micro-intervals is $m = 60$.

We consider $|Q| = 52$ feasible shift patterns. Each shift results in a working week for a courier made up of: 5 days per week for 8 hours per day, 4 days per week for 10 hours per day, or 3 days per week for 6 hours per day. The duration of each shift pattern determines its cost. For instance, shifts covering 40 hours per week are assigned a cost of 40, whereas the 18 hours shifts are assigned a cost of 18.

Demands are spatially independent and uniformly distributed over $Q = \{(o,d) \in \{1,\ldots,25\}^2 : o \neq d\}$. For each origin-destination pair, their arrival times constitute a Poisson process with rate $\lambda_{o,d}$ over the planning horizon $H$. The arrival rates $\lambda_{o,d}$ are set in such a way that the expected overall number of daily requests $\mu$ is 120 (or, equivalently, 600 weekly requests). Moreover, we assume that the inconvenience associated to each request is represented by its waiting time.

The number of vehicles in the initial solution is determined so to keep the average utilization factor $\rho$ of a vehicle lower than a given threshold $\alpha$. In our implementation $\alpha = 0.8$. The details of this procedure are as follows. We first randomly generate a number of shift patterns $q \in Q$ spanning all the planning horizon $H$, so that each micro-interval is covered by at least one shift. We refer to this set of randomly generated shift patterns as $Q'$. Then, we initially set $x_q = 1$ for each $q \in Q'$. Hence, in order to determine the minimum number of vehicles needed to have $\rho \leq \alpha$, we use the following formula for each micro-interval $h$:

$$\rho_h = \frac{\lambda_h \times 60'}{v_h} \leq \alpha \tag{5.21}$$

where $60'$ is the average service time estimated with reference to the service territory we are using for our experiments, $\lambda_h$ is the arrival rate of requests during micro-interval $h$, and $v_h$ is the minimum number of vehicles we are seeking. Thus, reversing equation (5.21) we obtain:

$$v_h \geq \frac{\lambda_h \times 60'}{\alpha}$$

Finally, we distribute these $v_h$ vehicles among all $q \in Q'$ such that $q$ covers micro-interval $I_h$.

Once obtained an initial solution, in order to assess the benefits that can be achieved using the ANE approximation, we use three different procedures. Anyway, before presenting them, we describe a method to recover feasibility for an infeasible solution, because it is used within the procedures described in the following subsections. We refer to this procedure as “Make Feasible” (MF in the following), and report the pseudo-code.

```
**Procedure** MF
**Input**: $x$
**while**($\exists j' : g_{j'}(x,\xi) > T_{j'}$) **do begin**
    Determine $q^* = \arg\min_{q \in \Omega_{j'}} \frac{c_q}{\sum_{j \in S_q} \max(0, g_j(x,\xi) - T_j)}$ where $\Omega_{j'}$ is the set of shift patterns covering at least partially QoS interval $j'$, and $S_q$ is the set of QoS intervals covered at least partially by $q$;
    Set $x_{q'} = x_{q'} + 1$;
    Update $g_j(x,\xi)$ through simulation;
**end**
**Output**: $x$
```

Now, we describe the procedures used to test the effectiveness of the ANE approach.

### 5.4.1 Careless GRASP

Our first GRASP implementation chooses the neighbor of the current solution $x^{(k)}$ by randomly sampling the neighborhood (without constructing it explicitly). Let this neighbor be $x^{(k+1)}$. Then, we simulate it until we can declare it feasible/infeasible with a confidence level of 95%. Thus, if $x^{(k+1)}$ is feasible and improves the best known solution, we update the current solution, and iterate. On the other hand, if $x^{(k+1)}$ is an improving solution, but it is infeasible, we recover feasibility by means of the MF procedure. Then, we set the new feasible solution as the current solution, and iterate. If, finally, $x^{(k+1)}$ is worse than the best known solution, we discard it and re-initialize the GRASP with a new initial solution determined as described in Section 5.4.

### 5.4.2 Careful GRASP

In the Careful GRASP implementation, given the current solution $x^{(k)}$, we explicitly determine all the neighbors. After constructing the neighborhood, we sort the solutions in $N(x^{(k)})$ according to their objective function value. Then, we simulate each solution starting from the more attractive in terms of objective function until we find a solution $x'^{(k)}$ which is feasible. Hence, if such a solution is better than the best known solution, we choose it as the new current solution, build the corresponding neighborhood and iterate. On the other hand, if $x'^{(k)}$ is worse than the best known solution, we discard it and randomly generate a new initial solution.

### 5.4.3 ANE GRASP

The ANE GRASP procedure uses the ANE approach to explore the search space efficiently. More specifically, we start from a randomly generated initial solution and find the best neighbor by solving the ANE model. Then, we simulate such a solution until it can be considered feasible or not at 95% confidence level. If it is infeasible, we recover feasibility by using the MF procedure, set the new feasible solution as the current solution, and iterate. Alternatively, if the simulated solution is declared feasible and improves the best known solution, we set it as the current solution and continue the search. On the other hand, if the solution is not better than the best known solution, we discard it and re-initialize the GRASP. We consider two variants of the ANE GRASP. A first variant computes the productivity using the naïve load assignment procedure, whereas the second variant determines such quantities using the fitting procedure.

### 5.4.4 Results

All the procedures were coded in C++ and use ILOG CPLEX 8.1 to solve the mathematical models underlying the ANE approach.

Our computational experiments consist of a dataset of 20 randomly generated instances. For all of the procedures we impose a time limit of 50,000 seconds. We do not report any result for the Careful GRASP, because this procedure is extremely slow and uses the whole time in order to find the initial solution, build the neighborhood, and sort the solutions. Thus, it never improves the initial solution.

Table 5.1 reports results for the Careless GRASP in which the distance between the current solution $x^{(k)}$ and the solutions of the neighborhood is 5, so that, according with the description of section 5.3, we set $|N(x^{(k)})| = 5$. The column headings are as follows:

- $INSTANCE$: progressive number of the randomly generated instance;
- $VIS\_SOL$: number of solutions visited throughout the GRASP;
- $RESTART$: number of local search restarts;
- $SAMPLES$: average number of samples needed to declare the feasibility or infeasibility of a solution at the 95% confidence level;
- $MF$: number of MF runs;
- $FEASIBLESOLS$: number of solutions which are declared feasible through simulation;
- $z$: objective function value of the first initial solution;
- $T_{SIM}$: average time (in seconds) needed to simulate a solution;
- $z^*$: best found solution.

Then, Table 5.2 summarize the results for the ANE GRASP with $|N(x^{(k)})| = 5$ and productivity computed through the naïve load assignment procedure. The column heading which has not been explained yet is as follows:

- *OBJDEV*: average deviation of the objective function taking the Careless GRASP as benchmark.

As data indicate, the ANE approximation, when combined with an intuitive procedure for the computation of a courier productivity, performs poorly. This results in solutions which, on the average, are 2% worse than a procedure which chooses the neighbor completely randomly.

Table 5.3 reports a similar comparison between the Careless GRASP and the ANE GRASP in which we use the fitting procedure to obtain the productivity. As a result, we obtain an average 7% improvement over the Careless GRASP. This behavior is explained with the better accuracy of the fitting procedure in the estimation of the productivity. Moreover, the fitting procedure produces estimates at no cost, because it takes only milliseconds to obtain the optimal solution of the corresponding model.

Our last computational experiment aims at verifying whether we can obtain a further improvement with a larger neighborhood size. Table 5.3 contains data for the comparison between the Careless GRASP and the ANE GRASP with $|N(x^{(k)})| = 10$ and the fitting procedure. The table shows that, when the neighborhood size increases, the ANE GRASP with the fitting procedure still outperforms the Careless GRASP, but its results worsen if compared to the ANE GRASP with a smaller neighborhood size.

## 5.5 Conclusions

In this chapter we have studied the same-day Courier Shift Scheduling Problem, a tactical problem which amounts to minimize the staffing cost subject to probabilistic service level requirements. We have devised an Approximate Neighborhood Evaluation mechanism in order to explore the search space efficiently. This model relies on the estimation (via simulation) of a reduced number of parameters, among which the productivity of a courier. With reference to this parameter we have also developed two different approaches for its estimation. The first approach is based on a naïve load assignment scheme, whereas the other procedure fits the ANE model in the current solution.

Results reported in Table 3.2 indicate that, in a typical same-day courier setting, the ANE approach combined with a simple load assignment procedure to determine the productivity performs poorly when compared to a random procedure for the choice of the neighbor.

On the other hand, as reported in Table 5.3, when using a fitting procedure to calculate the productivity, the ANE procedure outperforms a random procedure, resulting in an average cost reduction of 7%, which, in a multi-billion dollars market like that of couriers, can lead to millions-dollars of savings.

Finally, Table 5.4 shows that the benefits obtained through the ANE model decrease as the neighborhood size increases.

Table 5.1. Careless GRASP with a neighborhood size of 5.

| *INSTANCE* | *VIS_SOL* | *RESTART* | *SAMPLES* | *MF* | *FEASIBLESOLS* | $z$ | $T_{sim}$ | $z^*$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 505 | 248 | 6.89 | 4 | 236 | 1280 | 177.23 | 856 |
| 2 | 633 | 313 | 6.05 | 6 | 294 | 1228 | 140.07 | 914 |
| 3 | 625 | 302 | 5.98 | 4 | 293 | 1280 | 139.58 | 864 |
| 4 | 289 | 138 | 9.43 | 2 | 139 | 1264 | 318.19 | 896 |
| 5 | 242 | 242 | 6.67 | 6 | 241 | 980 | 174.66 | 908 |
| 6 | 489 | 249 | 6.96 | 8 | 227 | 1192 | 181.94 | 944 |
| 7 | 666 | 319 | 5.47 | 2 | 318 | 1200 | 134.26 | 896 |
| 8 | 136 | 66 | 13.56 | 2 | 59 | 1090 | 658.65 | 912 |
| 9 | 838 | 411 | 5.01 | 2 | 408 | 1520 | 106.19 | 916 |
| 10 | 427 | 211 | 8.56 | 4 | 199 | 1280 | 213.52 | 882 |
| 11 | 589 | 292 | 5.79 | 5 | 292 | 1490 | 152.89 | 936 |
| 12 | 583 | 290 | 5.47 | 5 | 288 | 1016 | 153.45 | 944 |
| 13 | 397 | 194 | 8.94 | 3 | 200 | 1220 | 233.38 | 882 |
| 14 | 477 | 220 | 6.82 | 3 | 254 | 1520 | 180.01 | 872 |
| 15 | 688 | 329 | 5.32 | 5 | 354 | 1388 | 126.41 | 882 |
| 16 | 532 | 315 | 9.98 | 3 | 286 | 1170 | 432.65 | 1122 |
| 17 | 532 | 257 | 6.01 | 6 | 269 | 944 | 162.46 | 896 |
| 18 | 689 | 328 | 5.62 | 6 | 355 | 1032 | 125.9 | 936 |
| 19 | 582 | 277 | 6.15 | 6 | 299 | 1200 | 149.39 | 882 |
| 20 | 540 | 260 | 6.34 | 7 | 273 | 1200 | 164.62 | 936 |

Table 5.2. ANE GRASP combined with the naïve load assignment procedure and a neighborhood size of 5

| *INSTANCE* | *VIS_SOL* | *RESTART* | *SAMPLES* | *MF* | *FEASIBLESOLS* | $z$ | $T_{sim}$ | $z^*$ | *OBJDEV* |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 315 | 254 | 7.25 | 5 | 264 | 1280 | 164.42 | 944 | 0.103 |
| 2 | 423 | 279 | 6.75 | 4 | 357 | 1228 | 121.14 | 944 | 0.033 |
| 3 | 486 | 336 | 5.67 | 4 | 362 | 1280 | 106.87 | 874 | 0.012 |
| 4 | 256 | 228 | 8.23 | 2 | 248 | 1264 | 196.3 | 936 | 0.045 |
| 5 | 472 | 347 | 5.70 | 5 | 436 | 980 | 109.42 | 944 | 0.040 |
| 6 | 370 | 338 | 7.03 | 5 | 334 | 1192 | 135.62 | 944 | 0.000 |
| 7 | 497 | 345 | 5.21 | 1 | 472 | 1200 | 103.53 | 856 | -0.045 |
| 8 | 172 | 155 | 8.32 | 1 | 163 | 1090 | 295.11 | 936 | 0.061 |
| 9 | 519 | 508 | 5.02 | 1 | 512 | 1520 | 94.35 | 936 | 0.022 |
| 10 | 282 | 255 | 5.32 | 4 | 262 | 1280 | 103.53 | 882 | 0.000 |
| 11 | 398 | 337 | 6.94 | 6 | 343 | 1490 | 128.76 | 936 | 0.000 |
| 12 | 366 | 335 | 7.12 | 2 | 343 | 1016 | 136.69 | 944 | 0.000 |
| 13 | 366 | 194 | 7.67 | 3 | 285 | 1220 | 147.11 | 882 | 0.000 |
| 14 | 363 | 265 | 6.02 | 2 | 339 | 1520 | 109.87 | 908 | 0.041 |
| 15 | 408 | 378 | 6.98 | 7 | 393 | 1388 | 132.54 | 882 | 0.000 |
| 16 | 356 | 234 | 9.54 | 3 | 186 | 1170 | 423.65 | 1122 | 0.000 |
| 17 | 158 | 135 | 7.89 | 4 | 148 | 944 | 150.85 | 944 | 0.054 |
| 18 | 464 | 349 | 5.93 | 5 | 434 | 1032 | 108.29 | 936 | 0.000 |
| 19 | 479 | 433 | 5.21 | 4 | 465 | 1200 | 102.42 | 882 | 0.000 |
| 20 | 363 | 260 | 7.53 | 6 | 332 | 1200 | 141.33 | 936 | 0.000 |
|  |  |  |  |  |  |  |  |  | 0.02 |

Table 5.3. ANE GRASP combined with the fitting procedure and a neighborhood size of 5.

| textitINSTANCE | *VIS_SOL* | *RESTART* | *SAMPLES* | *MF* | *FEASIBLESOLS* | $z$ | $T_{sim}$ | $z^*$ | *OBJDEV* |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 538 | 194 | 5.59 | 4 | 463 | 1280 | 103.28 | 850 | -0.100 |
| 2 | 475 | 184 | 5.73 | 5 | 240 | 1228 | 118.98 | 850 | -0.027 |
| 3 | 405 | 165 | 6.07 | 2 | 294 | 1280 | 135.4 | 882 | -0.043 |
| 4 | 245 | 98 | 7.42 | 3 | 165 | 1264 | 229.74 | 886 | -0.031 |
| 5 | 503 | 253 | 5.57 | 5 | 346 | 980 | 111.18 | 832 | -0.069 |
| 6 | 374 | 251 | 6.06 | 6 | 270 | 1192 | 139.98 | 944 | 0.121 |
| 7 | 570 | 214 | 5.07 | 0 | 406 | 1200 | 99.46 | 856 | 0.000 |
| 8 | 181 | 66 | 8.43 | 2 | 126 | 1090 | 313.15 | 802 | -0.067 |
| 9 | 446 | 166 | 5.85 | 4 | 315 | 1520 | 124.87 | 826 | -0.014 |
| 10 | 486 | 184 | 5.65 | 3 | 334 | 1280 | 116.64 | 834 | -0.026 |
| 11 | 246 | 93 | 7.21 | 1 | 173 | 1490 | 228.75 | 798 | -0.052 |
| 12 | 432 | 171 | 5.87 | 7 | 284 | 1016 | 132.48 | 878 | -0.013 |
| 13 | 381 | 194 | 6.45 | 3 | 260 | 1220 | 151.78 | 776 | -0.112 |
| 14 | 310 | 113 | 6.04 | 4 | 275 | 1520 | 132.40 | 864 | 0.049 |
| 15 | 394 | 154 | 6.89 | 5 | 342 | 1388 | 201.76 | 834 | -0.005 |
| 16 | 332 | 162 | 8.97 | 4 | 185 | 1170 | 385.98 | 990 | -0.059 |
| 17 | 354 | 290 | 6.10 | 7 | 337 | 944 | 139.76 | 856 | 0.000 |
| 18 | 329 | 122 | 6.87 | 2 | 288 | 1032 | 170.54 | 834 | -0.059 |
| 19 | 364 | 121 | 6.79 | 4 | 312 | 1200 | 149.06 | 852 | -0.009 |
| 20 | 324 | 260 | 6.75 | 6 | 298 | 1200 | 157.11 | 904 | 0.056 |
|  |  |  |  |  |  |  |  |  | -0.07 |

Table 5.4. ANE GRASP combined with the fitting procedure and a neighborhood size of 10.

| *INSTANCE* | *VIS_SOL* | *RESTART* | *SAMPLES* | *MF* | *FEASIBLESOLS* | $z$ | $T_{sim}$ | $z^*$ | *OBJDEV* |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 447 | 163 | 6.05 | 5 | 310 | 1280 | 121.59 | 886 | -0.100 |
| 2 | 330 | 130 | 7.94 | 5 | 213 | 1228 | 168.61 | 874 | -0.070 |
| 3 | 151 | 60 | 6.74 | 2 | 97 | 1280 | 132.65 | 922 | 0.021 |
| 4 | 245 | 89 | 9.32 | 2 | 163 | 1264 | 231.22 | 914 | -0.011 |
| 5 | 125 | 41 | 12.54 | 1 | 88 | 980 | 443.69 | 894 | -0.084 |
| 6 | 356 | 228 | 6.83 | 6 | 259 | 1192 | 140.96 | 842 | 0.000 |
| 7 | 559 | 182 | 5.45 | 3 | 391 | 1200 | 99.62 | 856 | -0.045 |
| 8 | 157 | 48 | 10.57 | 1 | 102 | 1090 | 369.96 | 860 | -0.121 |
| 9 | 117 | 51 | 12.89 | 3 | 74 | 1520 | 469.33 | 838 | -0.098 |
| 10 | 399 | 136 | 6.82 | 3 | 266 | 1280 | 139.99 | 856 | -0.054 |
| 11 | 439 | 189 | 6.21 | 5 | 379 | 1490 | 123.6 | 842 | -0.147 |
| 12 | 316 | 101 | 8.23 | 7 | 256 | 1016 | 175.96 | 890 | -0.070 |
| 13 | 534 | 172 | 5.78 | 4 | 364 | 1220 | 105.83 | 874 | -0.120 |
| 14 | 303 | 108 | 7.98 | 3 | 270 | 1520 | 173.05 | 824 | -0.009 |
| 15 | 346 | 135 | 7.01 | 3 | 300 | 1388 | 159.88 | 838 | -0.054 |
| 16 | 294 | 165 | 12.93 | 2 | 230 | 1170 | 523.04 | 1052 | -0.118 |
| 17 | 362 | 279 | 6.79 | 6 | 341 | 944 | 135.44 | 856 | -0.045 |
| 18 | 221 | 93 | 9.54 | 1 | 195 | 1032 | 250.11 | 886 | -0.109 |
| 19 | 380 | 112 | 6.91 | 3 | 325 | 1200 | 146.28 | 860 | -0.034 |
| 20 | 365 | 260 | 6.90 | 6 | 323 | 1200 | 145.52 | 856 | -0.034 |
|  |  |  |  |  |  |  |  |  | -0.02 |

# Chapter 6

# Conclusions and Future Work

In this thesis we discussed various aspects related to real-time fleet management. In this final chapter we give a short summary of the scientific contributions we believe have been provided in this thesis. Furthermore, we provide some directions for future work needed in order to extend the results in this thesis.

## 6.1 Conclusions

In this thesis we have first studied the Dynamic and Stochastic Traveling Salesman Problem and devised exact and heuristic waiting policies under the hypothesis that a probabilistic characterization of the customer requests is available. We have developed a Markov Decision Process as well as a lower bound based on the availability of perfect information. We have assessed the value of two waiting strategies against this lower bound.

In Chapter 3, we have devised a set of a priori routing schemes for use in a Dynamic Traveling Salesman environment. We have first described a formal dynamic programming formulation and presented the preliminary results. Then, we have discussed strategies for implementing a priori routes within this dynamic routing problem, with the aim to determine whether there is some value in devising involved strategies for the construction of the a priori routes. Results show that, when the number if late-request (dynamic) customers is 25% this effort is not worthwhile. On the other hand, when the number of late-request customers increases to 50%, more involved procedures offer improvement. In this case, a good choice for an a priori tour construction heuristic is a strategy which builds the a priori tour as a shortest path through both the advance- (static) and late-request customers.

In Chapter 4 we have considered the Dynamic and Stochastic Vehicle Dispatching Problem with Pickups and Deliveries, a problem faced by local area courier companies serving same-day pickup and delivery requests for the transport of letters and small parcels. We have proposed an anticipatory mechanism which we have then embedded in both an insertion and a local search algorithm. We tried to keep the computational effort low through two main features: firstly, the number of demand samples was determined through an IZS procedure that allows to eliminate, at an early stage of experimentation, those solutions that are clearly inferior; secondly, we limited the computational effort by applying the sampling procedure to a short-term horizon whose duration was empirically correlated to the problem data. Computational results have shown that the anticipatory insertion algorithm provides consistently better solutions than its reactive counterpart. They have also proved that, in a typical same-day courier setting, an anticipatory local search procedure allows to yield only very slight improvements on a simpler anticipatory insertion policy.

Finally, Chapter 5 introduced the same-day Courier Shift Scheduling Problem, a tactical problem which amounts to minimize the staffing cost subject to probabilistic service level requirements. We have devised an Approximate Neighborhood Evaluation (ANE) mechanism in order to explore the search space efficiently. This model relies on the estimation (via simulation) of a reduced number of parameters, among which the productivity of a courier. With reference to this parameter we have also developed two different approaches for its estimation. The first approach is based on an intuitive load assignment scheme, whereas the other procedure fits the ANE model in the current solution. Results indicated that, in a typical same-day courier setting, the ANE approach combined with a simple load assignment procedure to determine the productivity performs poorly when compared to a random procedure for the choice of the neighbor. On the other hand, when combining ANE with the fitting procedure, the ANE procedure outperforms a random procedure and we have also showed that the benefits decrease as the neighborhood size increases.

## 6.2 Future work

With respect to the DSTSP, the results we have obtained are based on a number of assumptions that should gradually be removed. Thus, future work should be aimed at removing: a) the hypothesis that request occurrence times $T_1 \leq T_2 \leq \ldots T_n$ are sorted in non-increasing order; b) the assumption that the order of service is given; c) the hypothesis that a customer request may arise at a single time instant. Moreover, in order to manage real-life instances, it would be interesting to develop a heuristic to handle as many customers as possible.

Chapter 3 evidenced that, when the number of dynamic customer is high, devising more involved a priori strategies can offer some value. Thus, in future work, we are interested in whether or not it is possible to construct good a priori routes such that, using waiting strategies, we are able to benefit from the ease of managing a priori routes without sacrificing the cost savings available in purely dynamic strategies. We are also interested in finding an efficient heuristic for the OPTW. The SC heuristic proposed in Section 3.4 is intuitively appealing, but requires an improved solution approach to truly assess its value.

In Chapter 4 an anticipatory mechanism for the dynamic and stochastic VDPPD was proposed. We have based our results on the hypothesis that the requests arrive according to a known stochastic process. Future work should be aimed at verifying how robust our approach is whereas demand sampling is based on an approximation of the arrival process.

Finally, in Chapter 5 we have introduced the same-day CSSP. Future work related to this problem should extend our approach to consider the case of multiple customer classes and multiple vehicle types.

# List of publications

In the following we report a list of papers which correspond to parts of this thesis, and have been submitted/accepted for publication in international journals.

1. G. Ghiani, G. Laporte, E. Manni & R. Musmanno. Waiting Strategies for the Dynamic and Stochastic Traveling Salesman Problem. *Submitted.* 2006.

2. G. Ghiani, E. Manni, A. Quaranta & C. Triki. Anticipatory Algorithms for the Dynamic and Stochastic Vehicle Dispatching Problem with Pickups and Deliveries. *Submitted.* 2007.

3. G. Ghiani, E. Manni & A. Quaranta. The Shift Scheduling in the Same-day Courier Industry. *Submitted.* 2007.

4. A. Attanasio, J. Bregman, G. Ghiani & E. Manni. Real-time Fleet Management at eCourier ltd. In *Dynamic Fleet Management - Concepts Systems, Algorithms & Case Studies.* V. Zeimpekis, C.D. Tarantilis, G.M. Giaglis & I. Minis (eds). pp 219–238. 2007.

# Bibliography

J. Atlason, M.A. Epelman, and S.G. Henderson. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research*, 127:333–358, 2004a.

J. Atlason, M.A. Epelman, and S.G. Henderson. Optimizing call center staffing using simulation and analytic center cutting plane methods. Technical Report IOE Technical Report 04-09, 2004b.

A. Attanasio, J. Bregman, G. Ghiani, and E. Manni. Real-time fleet management at eCourier ltd. In V.S. Zeimpekis, G.M. Giaglis, C.D. Tarantilis, and I.E. Minis, editors, *Dynamic Fleet Management - Concepts, Systems, Algorithms & Case Studies*, pages 236–258. Springer-Verlag, 2007.

A.N. Avramidis and P. L'Ecuyer. Modeling and simulation of call centers. In *Proceedings of the 2005 Winter Simulation Conference*, pages 144–152, 2005.

A.N. Avramidis, M. Gendreau, P. L'Ecuyer, and O. Pisacane. Agent scheduling in a multiskill call center. Technical Report CIRRELT 2007-44, Department of Computer Science, Strathclyde University, 2007.

R. Bent and P. Van Hentenryck. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52:977–987, 2004.

R. Bent and P. Van Hentenryck. Waiting and relocation strategies in online stochastic vehicle routing. In *Proceedings of the Twentieth International Joint Conference on Artifical Intelligence*, pages 1816–1821, 2007.

D.P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, Belmont, MA, 1995.

D.J. Bertsimas. *Probabilistic Combinatorial Optimizations Problems*. PhD thesis, Massachusetts Institute of Technology, 1988.

D.J. Bertsimas and G. Van Ryzin. A stochastic and dynamic vehicle routing problem in the euclidean plane. *Operations Research*, 39:601–615, 1991.

D.J. Bertsimas and G. Van Ryzin. Stochastic and dynamic vehicle routing problem in the euclidean plane with multiple capacitated vehicles. *Operations Research*, 41:60–76, 1993.

J.R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, New York, 1997.

J. Branke, M. Middendorf, G. Noeth, and M. Dessouky. Waiting strategies for dynamic vehicle routing. *Transportation Science*, 39:298–312, 2005.

L. Brotcorne, G. Laporte, and F. Semet. Ambulance location and relocation models. *European Journal of Operational Research*, 47:451–463, 2003.

G.G. Brown and G.W. Graves. Real-time dispatch of petroleum tank trucks. *Management Science*, 27:19–32, 1981.

G.G. Brown, C.J. Ellis, G.W. Graves, and D. Ronen. Real-time, wide area dispatch of mobil tank trucks. *Interfaces*, 17:107–120, 1987.

A.M. Campbell and B.W. Thomas. The probabilistic traveling salesman problem with deadlines. forthcoming in *Transportation Science*, 2007a.

A.M. Campbell and B.W. Thomas. Runtime reduction techniques for the probabilistic traveling salesman problem with deadlines. submitted for publication, 2007b.

A.M. Campbell and B.W. Thomas. Advances and challenges in a priori routing. In B. Golden, R. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and Challenges*. North-Holland, 2007 forthcoming.

A.M. Campbell, M. Savelsbergh, L. Clarke, and A. Kleywegt. The inventory routing problem. In T. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 95–112. Kluwer Academic Publishers, 1998.

Canadian Courier & Messenger Association. Industry overview, 2004.

A. Caprara, P. Toth, and M. Fischetti. Algorithms for the set covering problem. *Annals of Operations Research*, 98:353–371, 2000.

A. Caprara, L. Kroon, M. Monaci, M. Peeters, and P. Toth. Passenger railway optimization. In C. Barnhart and G. Laporte, editors, *Transportation*, Handbooks in OR & MS, pages 129–188. Elsevier Science, 2006.

F.W. Cathey and D.J. Dailey. A prescription for transit arrival/departure prediction using automatic vehicle location data. *Transportation Research Part C: Emerging Technologies*, 11:241–264, 2003.

J.-F. Cordeau and G. Laporte. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B*, 37:579–594, 2003.

J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M.M. Solomon, and F. Soumis. The vehicle routing problem with time windows. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, SIAM Monographs on Discrete Mathematics and Applications, pages 157–193. SIAM, Philadelphia, 2001.

J.-F. Cordeau, G. Ghiani, G. Laporte, and R. Musmanno. A parallel tabu search algorithm for the static dial-a-ride problem. Working Paper, 2002.

Y. Dumas, J. Desrosiers, E. Gelinas, and M.M. Solomon. An optimal algorithm for the traveling salesman problem with time windows. *Operations Research*, 43: 367–371, 1995.

T.A. Feo and M.G.C. Resende. Greedy randomized adpative search procedures. *Journal of Global Optimization*, 6:109–134, 1995.

M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98:23–46, 2004.

M. Fu. Optimization for simulation: theory vs. practice. *Informs Journal On Computing*, 14:192–215, 2002.

N. Gans, G. Koole, and A. Mandelbaum. Telephone call centers: Tutorial, review, and research prospects. *Manufacturing and Service Operations Management*, 5: 79–141, 2003.

M. Gendreau and J.-Y. Potvin. Dynamic vehicle routing and dispatching. In T.G. Crainic and G. Laporte, editors, *Fleet Management and logistics*, pages 115–126. Kluwer, Boston, 1998.

M. Gendreau, G. Laporte, and F. Semet. Solving an ambulance location model by tabu search. *Location Science*, 2:75–88, 1997.

M. Gendreau, F. Guerten, J.-Y. Potvin, and É. Taillard. Parallel tabu search for real-time vehicle routing and dispatching. *Transportation Science*, 33:381–390, 1999.

M. Gendreau, G. Laporte, and F. Semet. A dynamic model and parallel tabu heuristic for real-time ambulance relocation. *Parallel Computing*, 27:1641–1653, 2001.

M. Gendreau, F. Guerten, J.-Y. Potvin, and R. Séguin. Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. *Transportation Research Part C*, 14:157–174, 2006.

S. Genta and J.C. Muñoz. On assigning drivers for a home-delivery system on a performance basis. *Annals of Operations Research*, 155:107–117, 2007.

G. Ghiani, F. Guerriero, G. Laporte, and R. Musmanno. Real-time vehicle routing: Solution concepts, algorithms and parallel strategies. *European Journal of Operational Research*, 151:1–11, 2003.

G. Ghiani, E. Manni, A. Quaranta, and C. Triki. Anticipatory algorithms for the dynamic and stochastic vehicle dispatching problem with pickups and deliveries. Submitted, 2007.

G. Godfrey and W.B. Powell. An adaptive, dynamic programming algorithm for stochastic resource allocation problems i: Single period travel times. *Transportation Science*, 36:21–39, 2002.

M. Goetschalckx. A decision support system for dynamic truck dispatching. *International Journal of Physical Distribution and Materials Management*, 14:34–42, 1988.

B. Gopalakrishnan and E.L. Johnson. Airline crew scheduling: State of the art. *Annals of Operations Research*, 140:305–337, 2005.

F.S. Hillier and G.J. Lieberman. *Introduction to Operations Research*. McGraw-Hill, 2004.

L.M. Hvattum, A. Løkketangen, and G. Laporte. Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic. *Transportation Science*, 40(4):421–438, 2006.

S. Ichoua, M. Gendreau, and J.-Y. Potvin. Diversion issues in real-time vehicle dispatching. *Transportation Science*, 34:426–438, 2000.

S. Ichoua, M. Gendreau, and J.-Y. Potvin. Exploiting knowledge about future demands for real-time vehicle dispatching. *Transportation Science*, 40:211–225, 2006.

P. Jaillet. *Probabilistic Traveling Salesman Problems*. PhD thesis, Massachusetts Institute of Technology, 1985.

P. Jaillet. A priori solution of the traveling salesman problem in which a random subset of customers are visited. *Operations Research*, 36:929–936, 1988.

M.G. Kantor and M.B. Rosenwein. The orienteering problem with time windows. *Journal of the Operational Research Society*, 43:629–635, 1992.

P. Kilby, P. Prosser, and P. Shaw. Dynamic VRPs: A study of scenarios. Technical Report APES-06-1998, Department of Computer Science, Strathclyde University, 1998.

S. Kim, M.E. Lewis, and C.C. White III. Optimal vehicle routing with real-time traffic information. *IEEE Transactions on Intelligent Transportation Systems*, 6 (2):178–188, 2005.

S.-H. Kim and B.L. Nelson. A fully sequential procedure for indifference-zone selection in simulation. *ACM Transactions on Modeling and Computer Simulation*, 11:251–273, 2001.

S.-H. Kim and B.L. Nelson. Selecting the best system. In S.G. Henderson and B.L. Nelson, editors, *Handbooks in Operations Research and Management Science: Simulation*, pages 501–534. Elsevier, 2006.

G. Laporte and F.V. Louveaux. Solving stochastic routing problems. In T.G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 159–167. Kluwer, Boston, 1998.

G. Laporte and S. Martello. The selective traveling salesman problem. *Discrete Applied Mathematics*, 26:193–207, 1990.

G. Laporte, F.V. Louveaux, and H. Mercure. A priori optimization of the probabilistic traveling salesman problem. *Operations Research*, 42:543–549, 1994.

A. Larsen. *The Dynamic Vehicle Routing Problem.* PhD thesis, Technical University of Denmark, 2000.

A. Larsen, O.B.G. Madsen, and M.M. Solomon. Partially dynamic vehicle routing - models and algorithms. *Journal of the Operational Research Society*, 53:637–646, 2002.

A. Larsen, O.B.G. Madsen, and M.M. Solomon. The a-priori dynamic traveling salesman problem with time windows. *Transportation Science*, 38:459–472, 2004.

A. Larsen, O.B.G. Madsen, and M.M. Solomon. Recent developments in dynamic vehicle routing systems. In B. Golden, R. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges.* North-Holland, 2007, forthcoming.

A.M. Law and W.D. Kelton. *Simulation Modeling and Analysis.* McGraw-Hill, Boston, third edition, 2000.

K. Lund, O.B.G. Madsen, and J.M. Rygaard. Vehicle routing problems with varying degrees of dynamism. Technical report, Institute of Mathematical Modelling, Technical University of Denmark, 1996.

O.B.G. Madsen, H.F. Ravn, and J.M. Rygaard. A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives. *Annals of Operations Research*, 60:193–208, 1995a.

O.B.G. Madsen, K. Tosti, and J. Vælds. A heuristic method for dispatching repair men. *Annals of Operations Research*, 61:213–226, 1995b.

C. Malandraki and M.S. Daskin. Time dependent vehicle routing problems: formulations, properties, and heuristics algorithms. *Transportation Science*, 26:185–200, 1992.

Marketing Research for Industry. Courier services: UK, 2006.

Messenger Courier Association of the Americas. A profile of the less than 24 hour delivery industry and its value, 2006.

S. Mitrović-Minić and G. Laporte. Waiting strategies for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B*, 38:635–655, 2004.

S. Mitrović-Minić, R. Krishnamurti, and G. Laporte. Double-horizon based heurisitcs for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B*, 38:669–685, 2004.

N. Mladenović and P. Hansen. Variable neighbourhood search. *Computers & Operations Research*, 24:1097–1100, 1997.

J.D. Papastvrou. A stochastic and dynamic routing policy using branching processes with state dependent migration. *European Journal of Operational Research*, 95: 167–177, 1996.

J.-Y. Potvin, Y. Shen, and J.-M. Rousseau. Neural networks for automated vehicle dispatching. *Computers & Operations Research*, 19:267–276, 1992.

J.-Y. Potvin, G. Dufour, and J.-M. Rousseau. Learning vehicle dispatching with linear programming models. *Computers & Operations Research*, 20:371–380, 1993.

W.B. Powell. A stochastic model of the dynamic vehicle allocation problem. *Transportation Science*, 20:117–129, 1986.

W.B. Powell. Real-time optimization for truckload motor carriers. *OR/MS Today*, 18:28–33, 1990.

W.B. Powell and H. Topaloglu. Stochastic programming in transportation and logistics. In A. Ruszczynski and A. Shapiro, editors, *Handbooks in Operations Research and Management Sciences: Stochastic Programming*, pages 555–635. Elsevier, 2003.

W.B. Powell, Y. Sheffi, K.S. Nickerson, K. Butterbaugh, and S. Atherton. Maximizing profits for north american van lines' truckload division: A new framework for pricing and operations. *Interfaces*, 18:21–41, 1988.

W.B. Powell, P. Jaillet, and A. Odoni. Stochastic and dynamic networks and routing. In M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors, *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, pages 141–295. Elsevier Science, Amsterdam, 1995.

H.N. Psaraftis. A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transportation Science*, 14:130–154, 1980.

H.N. Psaraftis. Dynamic vehicle routing problems. In B.L. Golden and A.A. Assad, editors, *Vehicle Routing: Methods and Studies*, pages 223–248, Amsterdam, 1988. North-Holland.

H.N. Psaraftis. Dynamic vehicle routing: Status and prospects. *Annals of Operations Research*, 61:143–164, 1995.

M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, New York, 1994.

A.C. Regan, H.S. Mahmassani, and P. Jaillet. Improving efficiency of commercial vehicle operations using real-time information: Potential uses and assignment strategies. *Transportation Research Record*, 1493:188–198, 1994.

A.C. Regan, H.S. Mahmassani, and P. Jaillet. Dynamic decision making for commercial fleet operations using real-time information. *Transportation Research Record*, 1537:91–97, 1995.

C. Rego and C. Roucairol. Using tabu search for solving a dynamic multi-terminal truck dispatching problem. *European Journal of Operational Research*, 83:411–429, 1995.

S. Roy, J.-M. Rousseau, G. Lapalme, and J.A. Ferland. Routing and scheduling of transportation services for disabled: summary report. Technical Report, Transport Development Center, Montréal, Canada, 1984.

M. Savelsbergh and M. Sol. DRIVE: Dynamic Routing of Independent Vehicles. *Operations Research*, 46:474–490, 1998.

N. Secomandi. Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 27: 1201–1225, 2000.

N. Secomandi. A rollout policy for the vehicle routing policy with stochastic demands. *Operations Research*, 49:796–802, 2001.

L.I. Sennot. *Stochastic Dynamic Programming and the Control of Queueing Systems*. Wiley, New York, 1999.

J. Siek, L.-Q. Lee, and A. Lumsdaine. The boost graph library. `http://www.boost.org/libs/graph/doc/`, 2001.

M.M. Solomon. Algorithms for the vehicle routing and scheduling problems with time windows. *Operations Research*, 35:254–265, 1987.

M. Spivey and W.B. Powell. The dynamic assignment problem. *Transportation Science*, 38:399–419, 2004.

Statistics Canada. Couriers and local messengers industry, 2004.

M. Stefik. Planning with constraints (molgen: Part 1). *Artificial Intelligence*, 16: 111–139, 1981.

M.R. Swihart and J.D. Papastavrou. A stochastic and dynamic model for the single-vehicle pick-up and delivery problem. *European Journal of Operational Research*, 114:447–464, 1999.

B.W. Thomas. Waiting strategies for anticipating service requests from known customer locations. *Transportation Science*, 41(3):319–331, 2007.

B.W. Thomas and C.C. White III. Anticipatory route selection. *Transportation Science*, 38:473–487, 2004.

P. Toth and D. Vigo, editors. *The Vehicle Routing Problem.* SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia, 2002.

Transport Canada. Surface transportation policy, "canadian courier market size, structure and fleet analysis study", 2003.

J.I. van Hemert and J.A. La Poutré. Dynamic routing with fruitful regions: Models and evolutionary computation. In X. Yao, E. Burke, J.A. Lozano, J. Smith, J.J. Merelo-Guervós, J.A. Bullinaria, J. Rowe, P. Tino, A. Kabáan, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature VIII*, pages 690–699. Springer-Verlag, 2004.

A. Weintraub, J. Aboud, C. Fernandez, G. Laporte, and E. Ramirez. An emergency vehicle dispatching system for an electric utility in Chile. *Journal of the Operational Research Society*, 50:690–696, 1999.