Efficiencies in Freight & Passenger Routing & Scheduling

Final Report
METRANS Project
August, 2015

Principal Investigator:
Maged M. Dessouky

Ph.D. Graduate Student:
Han Zou

Daniel J. Epstein Department of Industrial and Systems Engineering
University of Southern California
Los Angeles, California

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, and California Department of Transportation in the interest of information exchange. The U.S. Government and California Department of Transportation assume no liability for the contents or use thereof. The contents do not necessarily reflect the official views or policies of the State of California or the Department of Transportation. This report does not constitute a standard, specification, or regulation.

Abstract

The problem we study concerns routing a fleet of capacitated vehicles in real time to collect shipment orders placed by a known set of customers. On each day of operation, only a subset of all customers request service. Some of these requests are known at the beginning of the day, while the rest arrive dynamically during the day. It is not known when and from which customers these dynamic requests may come from. An example of such an application is the daily operation of a trucking company that consolidates shipments from multiple suppliers.

The objective of this dynamic vehicle routing problem (DVRP) is three fold: first, to minimize the impact (on total travel distance) of knowing only partial information; second, to maximize the flexibility of the existing routes in a dynamic environment; third, to provide fast responses to dynamic customer requests. Thus, we propose an optimization-based, look-ahead dynamic vehicle routing framework that periodically re-optimizes the current vehicle routes by using both the known and forecasted information. Heuristic algorithms are designed to construct an initial solution, improve the initial solution, and adjust the waiting time along the vehicle routes.

We perform simulation experiments on well-known benchmark problem instances in the literature. For each instance, we compare the quality of our solution with other routing strategies. We see that the look-ahead routing strategy with forecasting of future requests outperforms a routing strategy that only makes use of the known demand information, in terms of total travel distance for instances with relatively fewer advance requests and more dynamic requests. Thus the look-ahead dynamic routing strategy shows its merits for problems with high level of uncertainty. Overall our proposed approach could generate routing solutions that could reduce freight vehicle miles traveled, thus minimizing the impact of freight on passenger travel since they primarily share the same road network, especially in major urban centers like Los Angeles.

Disclosure

The project was funded in entirety under this contract to California Department of Transportation.

Acknowledgement

We would like to thank METRANS for funding this research.

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1 Background

Many industries deal with the task of transporting massive amounts of goods over extended distances in a timely and cost-effective manner, including manufacturing, food, e-commerce, etc. Logistics has become the backbone that enables the productivity and mobility of these industries (Montreuil et al., 2013). Indeed, growth in the transportation sector recently has been on par with the growth in the Gross Domestic Product (GDP) in the US. According to statistics from the 2013 National Transportation Statistics report (LaHood and Porcari, 2013), expenditure on transportation activities amounted to 1,426 billion dollars in 2012, representing nearly 9 percent of the total US GDP. The Los Angeles region, in particular, has a large and diverse economic base, driven by large manufacturing, trade, and transportation sectors (Ang-Olson and Ostria, 2005). It is also ranked as one of the busiest freight centers in the US. Statistics show that in 2003, commodity flows into and out of the Los Angeles region totaled 588 million tonnage, of which trucking represented the biggest portion of 64 percent (Ang-Olson and Ostria, 2005).

However, the logistics sector as it is today functions in a way that is economically, environmentally, and socially unsustainable (Montreuil, 2011). For instance, the entire logistics sector is highly fragmented, with each supplier developing and operating its own distribution network that sees low capacity usage, high energy consumption, high greenhouse gas emission, and low workforce welfare (Montreuil, 2011). In order to compete effectively against their peers, companies have relied on internal optimization to reduce operating costs, but have overlooked opportunities for external consolidation. The increasing level of freight transportation also aggravates its impact on traffic congestion, and poses threats on the safety and efficiency of passenger traffic and other social functions that share the same road infrastructure. This phenomenon becomes more significant in densely populated urban areas, like Los Angeles.

External consolidation allows the sharing of vehicle capacity, delivery routes, and shipment orders among different suppliers or logistic service providers, thus creating a unified logistics network that sees increased capacity usage, reduced energy cost, pollution, and operating costs. A shared freight network also reduces the total truck miles, which in turn

reduces the usage of the road infrastructure that it shares with passenger traffic. Similarly, reduced freight traffic helps alleviate traffic congestions and the safety threat it poses on passenger traffic.

For example, the growth in the number of containers going through maritime container terminals has already introduced congestion and threatened the accessibility to these terminals. The congestion at a port, in turn, magnifies the congestion in the adjacent metropolitan traffic network. Besides, more and more container terminals require Just-In-Time (JIT) cargo delivery and pickup thanks to advances in information technologies, limited space for storing inbound and outbound containers, and increased competition. This constraint, together with others, demands higher service levels from trucking companies that fulfill transshipments of containers to and from the port. On the other hand, the highly competitive trucking industry is driven by the need to operate at the lowest possible cost, while satisfying consumer demand at the same time. These facts magnify the need for finding better ways of performing trucking operations in metropolitan areas adjacent to the ports. This problem falls in the scope of the Vehicle Routing Problem (VRP), which is formally defined as the problem of designing optimal routes of collection or delivery from one or several depots to a number of geographically dispersed customers.

In addition, logistic service providers (LSP) face uncertainties throughout various stages of their operations, including variable waiting and travel times due to traffic congestion, arrival of new orders, cancellation of existing orders, and unknown order sizes. These uncertainties limit the applicability of existing truck routing techniques, which have been primarily designed to solve the static routing problem where all information are perfectly known a priori. That is why, in the real world, human operators (dispatchers) still play a major role in the route planning and vehicle scheduling in the trucking industry (Flatberg et al., 2007; Laporte, 2009).

1.2 Problem Description

In this research, we aim to model and solve a truck routing problem that is representative of the daily operation of many logistic service providers, especially those who consolidate shipments from multiple suppliers. Suppose a trucking company operates a fleet of homogeneous vehicles to collect shipments from a known set of suppliers and transport the shipments back to a central depot. These suppliers can be seen as registered customers of the company. Their

locations and service time windows are known and fixed. However, each customer may not request service on each day. How often each customer requests service is determined by his/her own operation schedules, and can be seen as a given parameter in our problem. If a customer requests service, it can either do so at the beginning of the day (before the vehicles leave the depot), or at any time during the day. Customers who have requested service at the beginning of the day are called *advance* customers and must be serviced. All other customers, called *dynamic* customers, may potentially request service, but the company does not know whether and when they will do so. The company may have to reject a dynamic customer when he/she requests service if his/her shipment cannot be accommodated.

The situation described above can be modeled as a Dynamic Vehicle Routing Problem (DVRP). A DVRP is derived from a VRP when some information in the problem is revealed dynamically over time, instead of known before the vehicles are dispatched. In our problem, the set of customers needed to be serviced is random.

1.3 Motivation

The transportation industry, like many others, has undergone significant change in the last few years through the introduction of information technologies. Examples include:

- Vehicle tracking, such as global-positioning-systems (GPS), which allow vehicle locations to be determined with 3-meter level accuracy.
- Wireless communication, via satellite, cellular and paging networks, which enable 2-way communication with mobile fleets.
- Real-time information services, which allow for dynamic calculation of travel speeds.

Whereas in the past, it was difficult for a company to control or route vehicles once they left the terminal, these technologies make accurate dynamic real-time routing a very real possibility. Therefore, the dynamic VRP has emerged as an active and intense area of research, both due to industry needs, but also due to technological advances, including map databases, location determination technology (e.g., GPS), wireless communication and mobile computing.

In the vehicle routing problem, the customer demands, travel costs, and travel times are known in advance. The fundamental problem is to determine the optimal route that minimizes a certain objective such as fleet size and travel distance. Well-established routing and scheduling

algorithms that lead to optimum solutions have been presented in the literature. The built-in assumption of these approaches is that there will be small deviations on the realization of the demand and travel times from the plan so that the pre-determined routes form a basis for either the pickup or delivery schedule. However, in a highly dynamic and stochastic environment, the pre-planned optimal routes are no longer of practical use. In this case, most of the research effort has focused on easy to control dispatching rules in highly dynamic stochastic environments. The use of information technology in freight transportation systems has the potential to narrow the gap between highly uncertain systems in reality and the perfectly known static systems in theory.

Hence, on one end of the spectrum are the route planning techniques when it is reasonable to assume the system is deterministic and on the other end are the dispatching heuristics when the system is highly dynamic and uncertain. Therefore, there exists a gap in the literature for situations that are in between the two ends of the spectrum. To address this gap, there is a need to study the relationship between the uncertainties in the networks and the level of route planning in the freight transportation techniques.

The objective is to develop routing techniques that react better to uncertainties in demand to improve the operations of the trucking industry in terms of reducing vehicle miles, thus minimizing the impact of freight on passenger travel since they primarily share the same road network, especially in major urban centers like Los Angeles. We propose an optimization-based look-ahead vehicle routing framework that models and solves the dynamic vehicle routing problem by employing existing heuristics in the literature that are originally designed to solve static vehicle routing problems. In order to generate better solutions for problems with various degrees of uncertainty, the behavior of the proposed framework is tunable by adjusting different parameter settings of the framework and the heuristics.

1.4 Structure of the Report

The rest of the report is organized as follows. In Section 2, a literature review of the relevant problems is presented. Section 3 formally defines the problem and describes the look-ahead routing framework. In Section 4, all heuristic algorithms embedded in the solution framework are presented. Section 5 presents the experimental results of our solution framework

on a number of well-known benchmark problem instances in the literature. In Section 6, we discuss the implementation and applicability of our work. We conclude in Section 7.

2. Literature Review

In this section, we review the literature relevant to our research. The stochastic and dynamic vehicle routing problems have received a significant amount of attention in the literature during the last decade. In a stochastic vehicle routing problem (SVRP), not all information are known with certainty. Instead, only stochastic information is available for some elements of the problem. The randomness in the problem is usually realized before the vehicles are dispatched. However, in a dynamic vehicle routing problem (DVRP), uncertainties are revealed dynamically over time, during the planning horizon. Thus the current solution (vehicle routes) needs to be updated dynamically according to new information available in the system. The problem we study falls under the category of DVRP. We first review the studies on the stochastic and dynamic vehicle routing problems in general. We briefly introduce a classification system of various kinds of dynamic vehicle routing problems and their solution approaches and then review recent studies on solution approaches that are of the same category as ours.

2.1 Stochastic and Dynamic Vehicle Routing Problem

The problem we study falls under the category of DVRP. Dynamic vehicle routing problems arise naturally from a broad spectrum of real-world applications, including courier routing (Sungur et al., 2010), fleet configuration, inventory routing, Dial-a-Ride systems (Diana and Dessouky, 2004), etc. DVRP differentiates from the classic vehicle routing problem (VRP) in that some element of the problem is random and dynamic. Recent reviews on the DVRP are conducted by Pillac et al. (2013), by Laporte (2009), and by Bianchi (2000).

There are many potential sources of randomness in the DVRP. For example, the set of customers to serve may not be known with certainty; a customer may only request service by a given probability. Sometimes, the demand of a customer may not be known when the routing decisions have to be made. In addition, the cost matrix could also be stochastic, reflecting random travel times between locations due to varying traffic conditions. In a specific stochastic vehicle routing problem, randomness could come from a single source or multiple sources. Mix-and-match among potential sources of randomness makes the number of types of DVRP

problems grow exponentially. Therefore we briefly introduce a classification system of various problems and solution approaches presented in the literature.

We can classify a problem based on the following aspects.

- Whether a customer shows up or not
    - Deterministic
    - Stochastic
    - Dynamic
    - Partially dynamic
- Size of customer demand
    - Deterministic, known a priori
    - Known when service request is made
    - Known upon arrival at the customer
- Other sources of uncertainties
    - Stochastic customer location
    - Dynamic service time window
    - Stochastic service time
    - Stochastic travel time/cost

The most common source of randomness in a dynamic vehicle routing problem lies in whether a customer will show up or not, and the size of demand of each customer (Gendreau et al., 1996). These two sources of randomness are reflected in the first aspect stated above. For the first aspect, a problem may consist of all deterministic customers, stochastic customers (realized before vehicles are dispatched), dynamic customers (realized in real time), or partially dynamic customers (combination of deterministic and dynamic customers). For the second aspect, the demand of each customer may be deterministic (known a priori), known when service is requested, or known upon arrival at the customer. For the third aspect, possible sources of uncertainties not covered by the first two aspects include stochastic customer location, dynamic service time window, stochastic service time, stochastic travel time/cost, etc.

We can also classify a solution approach proposed for a dynamic vehicle routing problem. We introduce the following aspects.

- Modeling approach
    - Static routing

    - Reactive routing
    - Proactive routing
    - Local dynamic routing
    - Look-ahead dynamic routing
- Modeling technique
    - Robust optimization
    - Chance constrained model
    - Stochastic programming
    - Dynamic programming
    - Markov Decision Process (MDP)
    - Linear programming
- Solution technique
    - Exact algorithms
    - Heuristics

The modeling approach refers to the high level architecture of a solution. An approach is called static if vehicles follow a priori routes and take recourse actions when needed. It is called reactive routing if no planning is made and the decision maker relies solely on insertion and re-optimization when information in the system changes (Secomandi and Margot, 2009). Proactive routing describes the situation when strategic moves are made in anticipation of future system states. Such moves include preventive restocking, strategic waiting (Bent and Van Hentenryck, 2007; Branke et al., 2005; Thomas, 2007), etc. In a problem where new customer requests occur dynamically over time, route planning may choose to forecast future requests based on historical information. If they do so, the approach is called look-ahead dynamic. Rather, if no forecast is made, the approach is called local dynamic. The notion of "local dynamic" and "look-ahead dynamic" were first introduced by Chen and Xu (2006).

The modeling technique refers to the exact mathematical technique used to formulate the problem. Also, we can classify a solution approach based on the techniques used to solve the problem. Due to the complexity of dynamic vehicle routing problems, most of the techniques proposed in the literature fall into the heuristics category.

The problem we study can be classified as having partially dynamic customers, with demand known when service request is made, and no other uncertainties. The solution approach

we propose is a look-ahead dynamic, reactive vehicle routing approach. We use mixed integer programming to formulate the corresponding static vehicle routing problems, and use heuristic algorithms to solve them.

2.2 Periodic Re-optimization Approaches in DVRP

Since the vehicle routing problem has been extensively studied in the literature, one of the most intuitive approaches to solve a dynamic vehicle routing problem is to convert the dynamic problem into (possibly many) static problems and solve them sequentially. This approach is called re-optimization. Depending on how often static problems are constructed and solved, this approach involves into either the periodic re-optimization approach or the continuous re-optimization approach. The solution we propose adapts the periodic re-optimization approach. In particular, the planning horizon is divided into intervals of fixed length. Optimization of the current vehicle routes is conducted at the beginning/end of each time interval.

Psaraftis (1980) introduced periodic re-optimization using a dynamic programming approach. Chen and Xu (2006) considered a dynamic vehicle routing problem with hard time windows. They assumed that the dispatcher does not have any deterministic or probabilistic information on the location and the size of a customer order until it arrives. They embedded a dynamic column-generation-based algorithm into a periodic re-optimization framework. The approach showed its merits when compared with insertion-based heuristics on most problems.

Metaheuristics were developed to be combined with the periodic re-optimization framework. Montemanni et al. (2005) developed an Ant Colony System (ACS) to solve the vehicle routing problem with dynamic customers. One of the mechanisms of their solution is to hold dynamic customers that arrive within a time period until the end of that period. That is, customer requests are not handled immediately. This is not a desired feature of the solution we propose, because we want to provide dynamic customers with instant responses on whether they can be serviced or not.

Secomandi and Margot (2009) studied a vehicle routing problem with stochastic demands. The actual demand is only known when the vehicle arrives at the customer. The authors developed a finite-horizon Markov Decision Process (MDP) formulation for the single vehicle case. A partial re-optimization heuristic is proposed to solve the MDP. The authors compared

multiple heuristics to embed in the re-optimization framework. They argued that their best approach outperforms existing heuristics.

One major limitation of the re-optimization approach lies in the fact that all optimization needs to be performed before the decision maker can update each vehicle with its new route, potentially causing delays in routing operations (Pillac et al., 2013). The proposed approach mitigates the effect of this limitation by employing computationally fast heuristics and limiting the length of the look-ahead horizon. First, the construction, local search, and wait time adjustment heuristics largely use constant time calculations to reduce run time. Second, the solution framework uses an adjustable parameter to control the length of the look-ahead horizon, thus limiting the amount of computation needed at each decision epoch.

3. Solution Framework

In order to effectively model and solve this vehicle routing problem with dynamic customers, we setup multiple objectives to meet. Some of them will be explicitly included in the cost function, while others being implicitly designed into the solution framework. The objectives are:

- Minimize total travel distance.
- Minimize number of vehicles used.
- Maximize the flexibility of our solutions in a dynamic environment.
- Provide fast responses to the *dynamic* customers when they request service.

Thus we develop an optimization-based, look-ahead dynamic vehicle routing framework that instantly handles new information in the system and periodically re-optimizes current vehicle routes by using both realized and forecasted information. More specifically, we divide the planning horizon into time periods of equal length, and designate the beginning of each time period as a decision epoch. At each epoch, a static vehicle routing problem containing both know and forecasted information is solved. The solution is implemented according to pre-defined rules until the next decision epoch, or the end of the time horizon. In the following subsections, we first formally define the problem and then present the solution framework in detail by explaining the events in the dynamic environment.

3.1 Problem Definition

Suppose that we are making routing schedules for daily operations of a trucking company that collects shipments from a set of customers and transports them to a central depot. The planning horizon is one day, and is discretized into time steps. The length of the planning horizon is denoted as T_{max}. The company operates a fleet of homogeneous and capacitated vehicles. There are *N* potential customers (suppliers). Each customer has a fixed location, a known demand size, a known service time window and a service time of fixed length. Service time windows specify the earliest and latest times when service can be started at the corresponding customers. Each customer requests service at most once on each day. All these

information are deterministic and known a priori. The uncertainty lies in the fact not all customers request service every day. Some customers request service in advance, and are called *advance* customers. These requests are known at the beginning of the day. The rest of the customers are called *dynamic* customers, and they may or may not request service on that day. We assume that the probability a dynamic customer requests service is given (derived from historical information). The time when a dynamic customer requests service is called his/her request time. Request times are random variables.

We use the following notations. Generally, we use i, j to index customers, k to index vehicles, and t to index time.

Customers and requests:

N: total number of customers

$\mathbb{AC}$: set of advance customers

$\mathbb{DC}$: set of dynamic customers

d_i: demand of customer i

s_i: service time of customer i

r_i: request time of customer i

e_i: the earliest time that service can begin for customer i

l_i: the latest time that service can begin for customer i

Cost parameter:

$t_{i,j}$: minimum travel time between node i and j

Vehicles:

K: total number of customers

C: capacity of each vehicle

A realization of the dynamic vehicle routing problem (for a day) is fully determined by a set of advance customers and the request times of all dynamic customers who end up requesting service on that day. We assume that historical information is available concerning when each customer is likely to make a request. That is to say, each customer's request time is a random variable following a conditional probability distribution function $f_i(t)$ defined on the interval

$[0, e_i]$, meaning that each customer must make the request before the beginning of his/her service time window.

The company operates a centralized decision making unit with real-time two-way communication capability with all vehicles. At any point in time, the decision maker is aware of the state of each customer and the location and state of each vehicle. When a dynamic customer requests service, the decision maker instantly accepts or rejects the request based on feasibility. Feasibility of the problem is defined with respect to vehicle capacity and customer time window constraints.

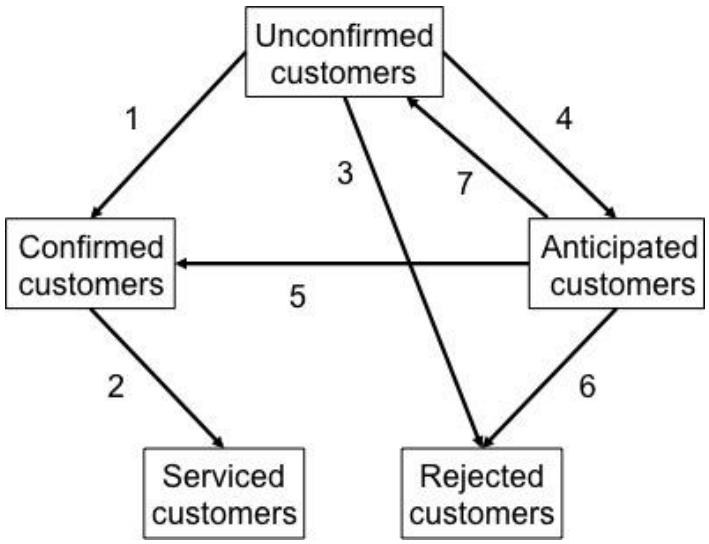
3.2 Customer States and Transitions

At any time instant t, each customer belongs to one and only one of the following five customer states. They are:

- Unconfirmed customer $\mathbb{U}(t)$. Customers who have yet to request service, and are not anticipated to request in the near future. At the beginning of the day, the set of unconfirmed customers initializes to the set of dynamic customers. $\mathbb{U}(0) = \mathbb{DC}$.
- Confirmed customer $\mathbb{C}(t)$. Customers who have requested service. The requests are confirmed (guaranteed to be serviced), but not yet serviced. At the beginning of the day, the set of confirmed customer initializes to the set of advance customers. $\mathbb{C}(0) = \mathbb{AC}$.
- Serviced customer $\mathbb{S}(t)$. Customers whose requests have been serviced.
- Rejected customer $\mathbb{R}(t)$. Customers who have requested service, but have been rejected.
- Anticipated customer $\mathbb{R}(t)$. Customers who have yet to request service, but are anticipated based on the forecast. Different forecast procedures are described in Section 3.4.1.

The state of each customer changes over time. State changes are triggered by certain dynamic events. The definition of all the events are given in Section 3.4. We now define transitions between customer states. Figure 3.1 illustrates all possible transition relationships between customer states.

Figure 3.1 Customer States and Transitions



We now explain each transition and specify the corresponding trigger event.

1. An unconfirmed customer becomes confirmed once he/she requests service and the request is accepted. Trigger event: new request.
2. A confirmed customer becomes serviced once a vehicle starts to travel to the customer. This vehicle is committed to reach the customer and collect the shipment from the customer. Equivalently speaking, we do not allow pre-emption in vehicle routes. Trigger event: vehicle departure.
3. An unconfirmed customer becomes rejected once he/she requests service and the request is rejected. Trigger event: new request.
4. An unconfirmed customer becomes anticipated if he/she has a "high chance" of making a request in the "near future". This kind of state transition is solely determined by the forecast procedure, which is embedded in the decision epoch. Trigger event: decision epoch.

5. An anticipated customer becomes confirmed once he/she requests service and the request is accepted. Note that the difference between type 5 and type 1 transitions lies in the initial state of the customer, equivalently, whether the customer is anticipated to make a request when he/she actually does. Trigger event: new request.
6. An anticipated customer becomes rejected once he/she requests service and the request is rejected. Similarly as above, the difference between type 6 and type 3 transitions lies in the initial state of the customer. Trigger event: new request.
7. An anticipated customer becomes unconfirmed if he/she does not make a request during the time period following the decision epoch at which he/she becomes anticipated. At the beginning of the next time period (a decision epoch), such customers are first moved back into the unconfirmed set. They may be anticipated again if they are selected again by the forecast scheme. The only exception is the last time period in the horizon, which is not followed by another decision epoch, since the end of horizon is reached. In this case, the difference between anticipated and unconfirmed states becomes irrelevant. Trigger event: decision epoch.

3.3 Partial Problem and Partial Solution

In a static vehicle routing problem, all information are perfectly known and are deterministic. The solution specifies a sequence of customer visits for each vehicle. In a dynamic vehicle routing problem, however, information are revealed gradually over time. The full realization of randomness cannot be known before the end of the day. Any time during the day, only partial information is known. We call it a partial vehicle routing problem P_t. The solution to a partial problem at time t is called a partial solution S_t. Partial solutions need to be constantly updated and modified according to new information revealed throughout the planning horizon. A partial solution is a collection of partial routes. $S_t = \{r_k\}$ where $k = 1, \dots, K$. A partial route consists of a sequence of customer locations $r_k = \{n_{0,k}, n_{1,k}, \dots, n_{|r_k|,k}, n_{|r_k|+1,k}\}$ where $|r_k|$ denotes the number of customers served by route k. Two special place holder nodes $n_{0,k}$ and $n_{|r_k|+1,k}$ are used to denote the starting and ending point of each route. $n_{|r_k|+1,k} = 0$ for all k and at all times because all routes terminate back to the depot. $n_{0,k}$ denotes the last customer that has

been served by the vehicle, which is also the starting position of this route. At $t = 0$, $n_{0,k} = 0$ for all k since all vehicle routes start at the depot at the beginning of the day.

In a dynamic context, the sequence of customer visits alone does not uniquely determine a vehicle route. In addition, we need to distribute the slack time along each vehicle route and specify the time when each vehicle arrives at and departs from each customer location. We use a_i and b_i to denote the arrival and departure times at customer i respectively. Different choices among ways to distribute the waiting time at each customer have shown to have an impact on the solution quality. We introduce two waiting time adjustment procedures, namely the **Push Backward** and **Push Forward** procedures. We will introduce them formally in Section 4.3.

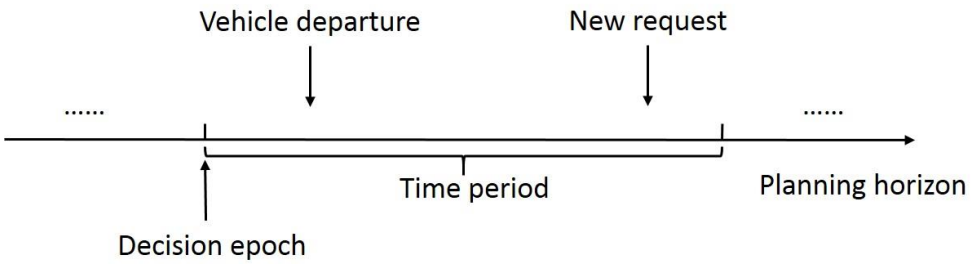
At any time t, the partial solution accommodates all confirmed customers $\mathbb{C}(t)$ and as many anticipated customers $\mathbb{A}(t)$ as possible. How partial solutions are constructed will be explained in Section 4.

3.4 Dynamic Events

In a dynamic vehicle routing environment, information is revealed gradually over time, instead of being known a priori. The central decision maker encounters various kinds of events throughout the planning horizon, including decision epoch, vehicle departure, and new request. It is important to note that "new request" is the only type of the above events outside the control of the decision maker. That is to say, a realization of the dynamic vehicle routing problem is fully determined by a set of advance customers $\mathbb{AC}$ and a series of new request events, each specifying the time of request of one dynamic customer in $\mathbb{DC}$. On the other hand, decision epochs and "vehicle departure" are events setup by the solution procedure that are used to implement the solution. The time when these events occur are fully determined by the solution procedure.

Figure 3.2 illustrates the time dynamic of the events. The horizontal axis refers to the planning horizon. Decision epochs are pre-designed into the time horizon and occur periodically at fixed intervals. Vehicle departure and new request events may occur at any time. We now explain what happens at each event in detail.

Figure 3.2 Time Dynamic of Events



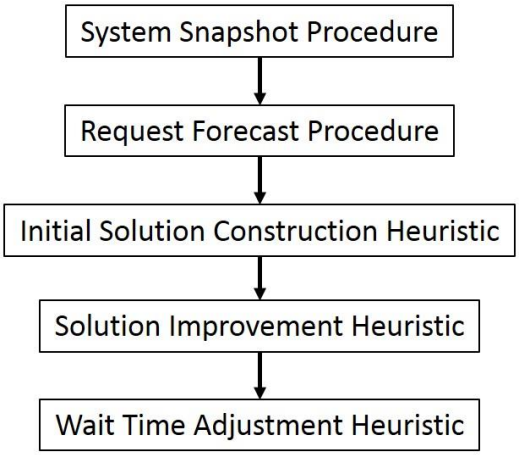
3.4.1 Decision Epoch

The decision epochs are the key component of the dynamic vehicle routing framework. The entire planning horizon is equally divided into M time periods. The beginning of each time period is called a decision epoch. Therefore the first decision epoch occurs at time 0. At each decision epoch, five solution procedures are called sequentially to construct and solve a deterministic vehicle routing problem. The procedures combine both the known information at the time, and forecasted information about the near future. The number of decision epochs ($numEpoch$) and the length of the forecast horizon ($forecastHorizon$) are parameters of the framework. We will discuss more about the solution parameters in Section 5.2.

Figure 3.3Figure 3.3 illustrates the solutions procedures called at each decision epoch. We now introduce the first two procedures and present the heuristics used for solution construction in Section 4.

The **System Snapshot Procedure** identifies the current the state of the problem and starts to construct a static vehicle routing problem. It first moves all customers in the anticipated set to the unconfirmed set. Those customers were anticipated at the last decision epoch, but have yet to request service. Time has elapsed and the probability that these customers will request service within the $forecastHorizon$ starting from the current time has changed. Thus we first empty the anticipated set and reconstruct it by using one of the forecast procedures.

Figure 3.3 Solution Procedures at Decision Epoch



The procedure then empties all the current partial vehicle routes. Those routes were constructed and optimized based on the information acquired and forecasted at the last decision epoch. They will be re-constructed and re-optimized to reflect the new information revealed in the system and updated forecast on future requests. There is one exception when removing customers scheduled on partial vehicle routes. If the first customer scheduled on the route is a confirmed customer, and the scheduled arrival time at the customer is within a pre-defined $tolerance$ amount from the current time, then the customer remains on the same route. This mechanism guarantees the feasibility of confirmed customers who have to be serviced soon from the current time. The complete process is shown below.

System Snapshot Procedure

Input: customer states, partial solution S_t

Output: updated customer states, updated partial solution

for all customers in $\mathbb{A}(t)$

remove from $\mathbb{A}(t)$ and insert into $\mathbb{U}(t)$;

for all non-empty vehicle routes r_k

if $n_{1,k} \in \mathbb{C}(t)$ ***and*** $a_{n_{1,k}} \leq currentTime + tolerance$

remove $n_{2,k}$ through $n_{|r_k|,k}$;

else remove $n_{1,k}$ through $n_{|r_k|,k}$;

Immediately following the system snapshot procedure is the request forecast procedure. This procedure exploits stochastic information on the customer request time to forecast future requests. We implement two basic forecast methods, namely threshold forecast and sampling forecast. Both methods depend on the parameter $forecastHorizon$, which specifies the number of time steps to look ahead. The threshold forecast procedure also depends on the parameter $threshold$, which is the smallest probability value above which the customer will be included in the anticipated set.

The rationale behind the **Threshold Forecast** is to calculate the probabilities that each unconfirmed will request service within the $forecastHorizon$, called p_i. We then include the customers whose probability of request is higher than a pre-defined threshold in the anticipated set. The higher the threshold, the fewer customers we anticipate, and thus the higher the quality of anticipation, because the chance that these anticipated customers will actually request service and become confirmed is higher.

Threshold Forecast

Input: customer states

Output: updated customer states

Let $t_1 = currentTime$ and $t_2 = currentTime + forecastHorizon$.

for each customer i in $\mathbb{U}(t)$

$p_i = \int_{t_1}^{t_2} f_i(t)dt$;

if $p_i \geq threshold$

remove i from $\mathbb{U}(t)$, insert i into $\mathbb{A}(t)$;

By anticipating all customers with high probabilities of making a request, the threshold procedure implicitly assumes that these customer will all make requests. However, the probability that all of these customers make requests is the product of probabilities that each customer makes a request, which is significantly lower than the probability that any customer makes a request. Thus the threshold procedure anticipates a scenario that has a much lower probability of happening.

Thus we have also developed the **Sampling Forecast**. We first calculate the probability that each unconfirmed customer will make a request within the $forecastHorizon$ in the same way as described above. Then we build a scenario consisting of a collection of new request events by sampling the joint probability distribution of all requests. By assuming that customers make requests independently from each other, we can simply sample each customer in sequence. The procedure is shown below.

Sampling Forecast

Input: customer states

Output: updated customer states

Let $t_1 = currentTime$, $t_2 = currentTime + forecastHorizon$.

for each customer i in $\mathbb{U}(t)$

$p_i = \int_{t_1}^{t_2} f_i(t)dt$;

sample $u \sim Uni(0,1)$;

if $u \leq p_i$

remove i from $\mathbb{U}(t)$, insert i into $\mathbb{A}(t)$;

3.4.2 Vehicle Departure

As stated in the previous sections, the solution to a dynamic vehicle routing problem needs to specify the arrival and departure times at each customer. Event "vehicle departure" refers to the incident that a vehicle leaves its current location for another location. These locations can be the depot or any customer location. Since we do not allow preemption of the vehicle routes (the vehicle is committed to service a customer once it starts to travel to that customer), a "vehicle departure" event triggers a state transition of the destination customer (from confirmed to serviced).

A partial solution accommodates both confirmed customers and anticipated customers. Thus there are two types of vehicle departure events. Those when the vehicle departs for a confirmed customer, and those when the vehicle departs for an anticipated customer. The following description shows what happens at a vehicle departure event.

Vehicle Departure Event

Input: customer states, partial solution S_t, vehicle k is about to leave for customer i

Output: updated customer states, current partial solution S_t

if $\mathrm{i} \in \mathbb{C}(\mathrm{t})$

remove i from $\mathbb{C}(\mathrm{t})$, insert i into $\mathbb{S}(\mathrm{t})$;

else

remove i from $\mathbb{A}(\mathrm{t})$, insert i into $\mathbb{U}(\mathrm{t})$;

Push Backward route k;

Push Forward route k;

The **Push Backward** and **Push Forward** algorithms are components of the waiting time adjustment heuristic. They are designed to distribute the slack time along vehicle routes. The **Push Backward** procedure tends to push all arrival and departure times towards to the beginning of the time horizon ($t = 0$), while the **Push Forward** procedure tends to push all arrival and departure times towards the end of the horizon ($t = T_{max}$). Both procedures will be described in detail in Section 4.3. It is important to point out that, by construction, vehicles do not travel to anticipated customers before they become confirmed. Thus, anticipated customers are moved into the unconfirmed set if they have not made a service request by the time the vehicle is ready to service them on the route.

3.4.3 New Request

Recall that a realization of the dynamic vehicle routing problem is fully determined by a set of advance customers and a series of new request events, each specifying the time of request of one dynamic customer. New request is the only type of dynamic event not controlled by the decision maker. Since each customer makes at most one request per day, new requests during the day can only come from either an unconfirmed customer, or an anticipated customer. The only

difference being that an unconfirmed customer does not exist in the current routes while an anticipated customer may have been scheduled into the current routes at the last decision epoch.

New Request Event

Input: customer states, partial solution S_t, new request from customer i

Output: updated customer states, current partial solution S_t

if i ∈ $\mathbb{A}$(t) ***and*** $i \in S_t$ (customer is routed)

remove i from $\mathbb{A}$(t), insert i into $\mathbb{C}$(t);

Push Backward route k;

Push Forward route k;

else if i ∈ $\mathbb{A}$(t) ***and*** $i \notin S_t$ (customer is not routed)

check feasibility of inserting customer i into S_t;

if feasible

insert i to the cheapest position using **Insert One Customer**;

remove i from $\mathbb{A}$(t), insert i into $\mathbb{C}$(t);

else remove i from $\mathbb{A}$(t), insert i into $\mathbb{R}$(t);

else (customer is not anticipated)

check feasibility of inserting customer i into S_t;

if feasible

insert i to the cheapest position using **Insert One Customer**;

remove i from $\mathbb{U}$(t), insert i into $\mathbb{C}$(t);

else remove i from $\mathbb{U}$(t), insert i into $\mathbb{R}$(t);

4. Heuristic

In this section we present the heuristics used to construct, improve, and update the partial solutions throughout the planning horizon.

4.1 Construction Heuristic

As described in Section 3.4.1, the construction heuristic is called at each decision epoch to construct an initial solution to the partial dynamic vehicle routing problem. The partial problem contains all confirmed (but yet to be serviced) customers $\mathbb{C}(t)$, and a set of anticipated customers $\mathbb{A}(t)$. All confirmed customers must be routed, while anticipated customers are routed until it is no longer feasible to schedule them on the existing routes. We develop an insertion-based heuristic that constructs all vehicle routes in parallel. This construction heuristic aims to minimize both total travel distance and the number of vehicles used. At each iteration, we calculate an impact measure of inserting each candidate customer into each feasible position in the partial solution. The measures are sorted. The customer with the lowest impact measure is inserted into the corresponding cheapest position.

We hereby introduce the general version of the heuristic as Algorithm 1. We use "candidate customers" to denote the set of all customers needed to be routed.

Algorithm 1: Construction Heuristic

Input: candidate customers CC, partial solution S_t

Output: updated partial solution S_t

while $CC \neq \emptyset$

  for each customer $i \in CC$

    for each vehicle route $r_k \in S_t$

      for each feasible insertion position $(n_{j-1,k}, n_{j,k}) \in r_k$

        calculate $impact(i, r_k, n_{j-1,k}, n_{j,k})$;

  sort $impact(i, r_k, n_{j-1,k}, n_{j,k})$ and find minimum;

insert i into r_k^* at position $\left(n_{j-1,k}^*, n_{j,k}^*\right)$, corresponding to the minimum impact;
remove i from CC;

$impact(i, r_k, n_{j-1,k}, n_{j,k})$ is a weighted average of both direct marginal travel distance and other surrogate cost measures of inserting customer i into route r_k at position $\left(n_{j-1,k}, n_{j,k}\right), 0 \leq j \leq |r_k| + 1$. Initial work on such impact measurements are presented by Solomon (1987) and Ioannou et al. (2001). We first present components of the impact measure, namely the self-impact SI, external-impact EI, and internal-impact II.

1. $SI(i, r_k, n_{j-1,k}, n_{j,k})$ denotes self-impact. Let $a_i\left(n_{j-1,k}, n_{j,k}\right)$ denote the arrival time at customer i if it is inserted into position $\left(n_{j-1,k}, n_{j,k}\right)$ on route r_k. Then the self-impact is calculated as the difference between the arrival time at customer i and the earliest time to start service at this customer. Self-impact measures the coverage of the service time window of customer i. A smaller value means extra slack time available for the insertion of an additional customer before or after i on route r_k.

$$a_i\left(n_{j-1,k}, n_{j,k}\right) = \max(e_i, a_{n_{j-1,k}} + s_{n_{j-1,k}} + t_{n_{j-1,k},i})$$

$$SI\left(i, r_k, n_{j-1,k}, n_{j,k}\right) = a_i\left(n_{j-1,k}, n_{j,k}\right) - e_i$$

2. $EI(i)$ denotes external-impact. It measures the impact of inserting customer i on other un-routed candidate customers $CC\backslash\{i\}$. Whenever we schedule an additional customer on the current vehicle routes, it becomes more difficult to schedule other customers due to potential time window infeasibility. The external-impact measures the amount of time window overlap between customer i and all other un-routed customers. A smaller external-impact value means less overlap between the time windows, and less of an impact on the un-routed customers. The value of external-impact does not depend on the position of insertion. Rather, it only depends on the customer we insert, and the current set of un-routed customers.

$$EI(i) = \frac{1}{|CC| - 1} \sum_{j \in CC\backslash\{i\}} \max\{\left(l_j - e_i - t_{i,j}\right), (l_i - e_j - t_{i,j})\}$$

3. $II(i, r_k, n_{j-1,k}, n_{j,k})$ denotes internal-impact. We first introduce three cost measures used to calculate internal-impact. The first measure c_1 is calculated as the marginal travel distance of inserting customer i. The second cost measure c_2 is calculated as the delay in arrival time at customer $n_{j,k}$ caused by inserting customer i right before it. This is the maximum amount of delay that will propagate along the route to all the customers following $n_{j,k}$. The lower the delay, the less of an impact the insertion has on existing customers on the route. The third measure c_3 is calculated as the time gap between the earliest possible arrival time at customer i and the latest possible time that service can start at customer i. This measure expresses the compatibility of the selected customer with the specific insertion position. It is evident that a smaller time gap means a more compact route. Which is generally preferred because we want to limit the number of vehicles used. Lastly, the internal-impact is calculated as a weighted average of all these cost measures by using parameters $\beta_1 + \beta_2 + \beta_3 = 1, \beta_1, \beta_2, \beta_3 \geq 0$.

$$c_1(i, r_k, n_{j-1,k}, n_{j,k}) = t_{n_{j-1,k},i} + t_{i,n_{j,k}} - t_{n_{j-1,k},n_{j,k}}$$

$$c_2(i, r_k, n_{j-1,k}, n_{j,k}) = \max\left(e_{n_{j,k}}, a_i + s_i + t_{i,n_{j,k}}\right) + \max(e_{n_{j,k}}, a_{n_{j-1,k}} + s_{n_{j-1,k}} + t_{n_{j-1,k},n_{j,k}})$$

$$c_3(i, r_k, n_{j-1,k}, n_{j,k}) = l_i - (a_{n_{j-1,k}} + s_{n_{j-1,k}} + t_{n_{j-1,k},i})$$

$$II(i, r_k, n_{j-1,k}, n_{j,k}) = \beta_1 c_1(i, r_k, n_{j-1,k}, n_{j,k}) + \beta_2 c_2(i, r_k, n_{j-1,k}, n_{j,k}) + \beta_3 c_3(i, r_k, n_{j-1,k}, n_{j,k})$$

Finally, the overall $impact(i, r_k, n_{j-1}, n_j)$ is calculated as a weighted average of self-impact, external-impact, and internal-impact with parameters $\alpha_{SI} + \alpha_{EI} + \alpha_{II} = 1, \alpha_{SI}, \alpha_{EI}, \alpha_{II} \geq 0$.

$$impact(i, r_k, n_{j-1}, n_j) = \alpha_{SI} SI(i, r_k, n_{j-1}, n_j) + \alpha_{EI} EI(i) + \alpha_{II}\, II(i, r_k, n_{j-1}, n_j)$$

Another insertion procedure similar to the one described above is developed to insert only one customer into the current vehicle route. This procedure is called during the **New Request Event**.

Algorithm 2: Insert One Customer

Input: customer i, partial solution S_t

Output: updated partial solution S_t

for each vehicle route $r_k \in S_t$

    for each feasible insertion position $(n_{j-1,k}, n_{j,k}) \in r_k$

        calculate $impact(i, r_k, n_{j-1,k}, n_{j,k})$;

sort $impact(i, r_k, n_{j-1,k}, n_{j,k})$ and find minimum;

insert i into r_k^* at position $(n_{j-1,k}^*, n_{j,k}^*)$, corresponding to the minimum impact;

4.2 Local Search

As described above, at each decision epoch, the construction heuristic is first called to build an initial partial solution serving all confirmed customers, and as many anticipated customer as possible. A local search heuristic follows to improve the initial solution. Since this heuristic is called during the day, while vehicles are traveling and serving customers, it has to be efficient. Thus we need to develop a local search procedure that balances between performance and speed.

The local search procedure we develop consists of two search operators (moves) and a simulated annealing-like mechanism to help the search escape from a local optimum. Simulated Annealing (SA) is a metaheuristic first proposed by Kirkpatrick (1983). The algorithm has then been studied extensively and adapted to solve various problems. Early attempts to use simulated annealing in solving vehicle routing problems are presented by Teodorović and Goran (1992), and by Osman (1993). Since the construction heuristic tries to minimize the total travel distance and the number of vehicles used, all the routes in the initial solution are fairly packed with customers. Besides, tight and hard time window constraints limit the neighborhood structure of the intra-route search operators. Thus both search operators we select are inter-route operators, namely the relocation operator, and cross operator.

The relocation operator is originally proposed by Savelsbergh (1992) for the classical VRP. This operator randomly selects two routes r_1, r_2 (r_1 must be non-empty) and a customer

$n_i \in r_1$. It then tries to relocate n_i to a random location on r_2. If both new routes are feasible, the move is accepted with a certain probability depending on the acceptance rule of the simulated annealing-like mechanism.

The cross operator is originally proposed by Savelsbergh (1992) and studied by many researchers including Potvin and Rousseau (1995). This operator randomly selects two non-empty routes r_1, r_2 and one customer on each route $n_i \in r_1, n_j \in r_2$. It splits routes r_1, r_2 at the position immediately following n_i and n_j respectively. Then two new routes are constructed by switching the slipped segments of the routes. If both new routes are feasible, the move is accepted with a certain probability depending on the acceptance rule of the simulated annealing-like mechanism.

Algorithm 3 shows how these two operators are incorporated in the simulated annealing-like mechanism. $maxIteration$ is a solution parameter that limits the number of iterations to perform for each search operator. In order to control the maximum amount of time used by the local search heuristic, we use the maximum number of iterations as the stopping criteria, instead of using the convergence rate.

Algorithm 3: Local Search

Input: partial solution S_t

Output: improved partial solution S_t

$counter = 1$;

while $counter \leq maxIteration$

$temperature = \frac{maxIteration - counter}{maxIteration}$;

$cost =$ total cost of S_t;

perform search operator, get S_t';

$cost' =$ total cost of S_t';

if $acceptanceRule(cost, cost', temperature)$ == TRUE

$S_t = S_t'$;

$counter++$;

4.3 Waiting Time Adjustment

As mentioned before, a sequence of customer locations alone does not uniquely determine a solution in the dynamic vehicle routing environment, due to possible slack time along the route. We need to specify the time when each vehicle arrives at and departs from each customer location. Based on the same sequence of consumer locations, how to distribute the slack time among all the customers on the route is a key factor that affects the final cost of a solution. Indeed, waiting is shown to be a useful strategy in handling dynamic customer arrivals especially those with time windows (Pillac et al., 2013).

We develop a waiting time adjustment scheme that aims to increase the flexibility of the partial vehicle routes in light of possible new customer requests. The general idea is that vehicles should wait at their current customer locations after service, and travel to the next customer at the earliest time to ensure no waiting time at the next customer. Doing so allows vehicles extra time waiting for new information to be revealed, instead of being pre-maturely committed to the next customer (because we do not allow pre-emption). This waiting time adjustment scheme is called **Push Backward**.

By default, we set all arrival and departure times based on wait-first strategy. That is, all vehicles wait at their current customer location after service (if necessary), and travel to their next customer at the earliest time to ensure no waiting time at the next customer. Equivalently speaking, the arrival time at each customer is always the same as the time when service starts at the customer. For simplicity, we use i to denote a customer, i^- and i^+ to denote the predecessor and successor of that customer on the route containing i. Then this waiting strategy can be represented as follows.

$$a_i = \max\{e_i, a_{i^-} + s_{i^-} + t_{i^-,i}\}$$

$$b_i = a_{i^+} - t_{i,i^+}$$

The arrival time at customer i is the maximum of the beginning of his/her service time window and the earliest time that the vehicle can arrive. The time when the vehicle departs from customer i is set backward based on the time that the vehicle arrives at his/her successor.

Algorithm 4: Push Backward

Input: partial solution S_t

Output: waiting time adjusted partial solution S_t

for each route $r_k \in S_t$

for $i = 1, i \leq |r_k|$

$a_{n_i} = \max\{e_{n_i}, a_{n_{i-1}} + s_{n_{i-1}} + t_{n_{i-1},n_i}\}$;

$b_{n_{i-1}} = a_{n_i} - t_{n_{i-1},n_i}$;

$++i$;

We have also developed another waiting time adjustment scheme that tries to increase the chance of accommodating anticipated customers when they actually make a service request. Recall that at each decision epoch, a new set of anticipated customers is constructed and scheduled in the partial vehicle routes. These customers remain in the anticipated set until either when the next decision epoch is reached (at which time they become unconfirmed) or when the customer requests service (at which time they become either confirmed or rejected), whichever comes first. So it is preferred that anticipated customers can wait as long as possible before letting the vehicle drop them from the route, since vehicles do not travel to anticipated customers before they become confirmed. In particular, we work from the end of the route to the front, and try to push all arrival and departure times as much towards to the end of the horizon as possible. We do so for every customer until (and including) the first anticipated customer on the route (counting from the front). Thus we have created the largest slack time right before the first anticipated customer on the route. This slack time allows the maximum amount of time for the vehicle to wait for the anticipated customer to request service, while maintaining time window feasibility for all the other customers.

Algorithm 4: Push Forward

Input: partial solution S_t

Output: waiting time adjusted partial solution S_t

for each route $r_k \in S_t$

if r_k contains at least one anticipated customer

let n_{i^*} denote the first anticipated customer on r_k

for $i = |r_k|, i \geq i^*$

$a_{n_i} = \min\{l_{n_i}, a_{n_{i+1}} - t_{n_i,n_{i+1}}\}$;

$b_{n_i} = a_{n_{i+1}} - t_{n_i,n_{i+1}}$;

$i - -$;

5. Experimental Results

5.1 Problem Instances and Realizations

We test our model and solution approach on a modified Solomon (1987) vehicle routing problem with time windows (VRPTW) instance. We call Solomon instances *base instances* of our experiments. A *base instance* specifies the set of all customers with their locations, demands, and time windows. It also specifies the length of the planning horizon, the maximum number of vehicles available, and the capacity of each vehicle. We transform a base instance into a *DVRP instance* within the dynamic context by specifying two parameters, namely the percentage of advance customers ($ACPercent$), and the probability that a dynamic customer makes a request ($RequestProb$). Therefore the combination of a base instance with two instance parameters uniquely determines a DVRP instance. In addition, given that customer i is to make a request, his/her request time is defined by the conditional probability density function $f_i(t)$. We use triangular distribution functions to model $f_i(t)$. In particular, the minimum value of the distribution is set to 0. The maximum value of the distribution is set to be equal to the beginning of the time window of each customer, e_i. The mode of the distribution is set to $\frac{e_i}{4}$. By this construction, the dynamic customers will always request service between the beginning of the day and the time when their service time windows begin. They have higher chances of making requests early within the time frame. This setup tends to make the dynamic vehicle routing problem more feasible, since it gives more time to schedule the request when a dynamic customer makes a request.

A DVRP instance constructed as above specifies all deterministic and stochastic information of the problem. A *realization* of the problem specifies the set of advance customers, a subset of dynamic customers who are to make a request, and the precise request times of these dynamic customers. A *realization* reflects the actual routing problem faced by decision makers, and is constructed in the following steps.

Construct Realization

Input: DVRP instance

Output: realization

Let $numAC = N * ACPercent$;
Let $perm$ be a random permutation of the sequence $(1, \dots, N)$;
for $i = 1, \dots, numAC$
    insert customer $perm(i)$ into $\mathbb{AC}$;
for each customer $i, i \notin \mathbb{AC}$
    sample $u \sim Uni(0,1)$;
    if $u \leq RequestProb$
        sample $r_i \sim f_i(t)$;
        construct a "new request" event for customer i at time r_i;

We will build multiple DVRP instances corresponding to different value combinations of instance parameters. In order to reduce the variability of performance of the different solution approaches, each instance is replicated twenty times (twenty realizations), and the average result is reported.

5.2 Solution Parameters and Measurements

There are multiple solution parameters we can adjust to modify the behavior of the proposed look-ahead dynamic vehicle routing framework. These parameters are: the number of decision epochs $numEpoch$, the length of forecast horizon $forecastHorizon$, the mechanism used to construct the anticipated set of customers (either threshold or sampling forecast), and the parameters used in the heuristic algorithm. After running preliminary experiments, we have fixed the values of some parameters to levels that consistently generate better performance. In particular, we have set $numEpoch = 10$, $\alpha_{SI} = 0.33, \alpha_{EI} = 0.33, \alpha_{II} = 0.34, \beta_1 = 0.8, \beta_2 = 0.1$, and $\beta_3 = 0.1$. We test all other parameters and present the behavior of the solution framework in the next subsection.

For each realization of the problem, we compare the following three routing strategies in terms of three performance measurements, namely the total travel distance, the total number of

vehicles used, and the total number of rejected customers. We now introduce the routing strategies and the measurements we use.

1. Static routing: a static vehicle routing problem with time windows is built based on the realization, then solved by using a static VRP algorithm. In particular, the set of advance customers and the subset of dynamic customers who end up making a request are combined. It is then assumed that all of these customer requests are known at the beginning of the day and must be served. We solve this classic VRP by applying **Algorithm 1**, followed by **Algorithm 3**, which runs the cross operator for 5000 iterations, and then the relocation operator for another 5000 iterations. We record the total distance $static\_Distance$ and number of vehicles used $static\_NumV$. When using the static routing strategy, the number of rejected customers is always 0. This routing strategy serves as the base for comparison. The final solution generated by this strategy is called the static routing solution.
2. Look-ahead dynamic routing: the look-ahead vehicle routing framework is applied on the realization of the problem. Ten decision epochs are built into the planning horizon. At each epoch, **Algorithm 1** is first called to route only the confirmed customers. Then the algorithm is called a second time to route the anticipated customers. **Algorithm 3** follows to improve the initial solution. To ensure a fair comparison with the static routing strategy, we run each local search operator for 500 iterations at each decision epoch, such that the total number of iterations is the same as in the static strategy. Whenever a dynamic customer requests service, the procedure **New Request Event** is called to handle the request.

   We start this approach by imposing an extra constraint that the total number of vehicles used at any point in time does not exceed $staticNumV$. Doing so may lead to infeasibility of the problem (not all confirmed customer can be routed) or rejection of customer requests (a dynamic customer cannot be accommodated when he/she requests service). Thus we adapt a trial-and-error procedure to find the minimum number of vehicles that will ensure feasibility and no rejection of customer requests. In particular, we start by setting the number of vehicles allowed equal to $staticNumV$, and gradually increase this number until the dynamic routing strategy generates a feasible solution without any rejections. We call this final solution the

look-ahead dynamic solution. For this strategy, we record the number of rejected customers, $dynamic\_Rejected$, when using the $static\_NumV$ vehicles. We also record the $dynamic\_NumV$ number of vehicles that ensures no rejection of customer requests in the look-ahead dynamic routing solution and the corresponding total travel distance $dynamic\_Distance$.

3. Advance routing: this strategy makes no forecast and thus no planning on dynamic customers. The vehicle routes are constructed based on only the known demand. In particular, we apply **Algorithm 1** and **Algorithm 3** on only the set of advance customers. Each operator runs for 5000 iterations. No more construction and local search heuristics are called during the day. Whenever a dynamic customer requests service, the procedure **New Request Event** is called to handle the request. A similar trial-and-error procedure as described above is used. The final solution that ensures feasibility and no request rejection is called the advance routing solution. For this strategy, we record the number of rejected customers, $adv\_Rejected$, when using the $static\_NumV$ vehicles. We also record the adv_NumV number of vehicles that ensures no rejection of customer requests in the advance routing solution and the corresponding total distance $adv\_Distance$.

5.3 Simulations and Results

All the experimental results in this section used the Solomon (1987) instance R110 as the *base instance*. In the R110 base instance, all customers are randomly located around a central depot. None of the customer service time windows starts at time 0. This allows us to define the conditional probability functions of request times as described in Section 5.1. The length of the planning horizon is 230 time steps ($T_{max} = 230$). Since we have set $numEpoch = 10$, the length of each time period is 23 time steps. When testing the look-ahead dynamic routing strategy, we set the value of $forecastHorizon$ to be roughly equal to multiples of the length of a time period. We constructed various *DVRP instances* by using different value combinations of $ACPercent$, and $RequestProb$. The tested levels are $ACPercent = 0.1, 0.25, 0.5, 0.75$, and $RequestProb = 0.25,0.5,0.75$. So there are a total of 12 *DVRP instances.*

For each instance, we generate 20 realizations. We then perform the three routing strategies described above on each realization. The average results of the 20 realizations are presented below. In all the tables, we use heading "Dist %" to denote the percentage of total travel distance penalty of the strategy shown, as compared to the static routing strategy. We use heading "Vehicle" to denote the number of extra vehicles used by the strategy shown when compared to the static routing strategy. The heading "Rejected" denotes the number of rejected customers if using the same number of vehicles as in the static routing solution. It is important to point out that the static routing strategy assumes that all problem information are known in advance. In particular, it assumes that all dynamic customers who end up making a service request are known at the beginning of the day, so that they can be treated as advance customers. As a result, the static routing solution is generated based on perfect information of the problem, which by construction should out-perform the other two routing strategies proposed. Since the static routing solution is used as the base of comparison in all reported results, all performance measures are positive. It is also evident that lower values of all measures indicate a better performance.

5.3.1 Advance Routing

Table 5.1 shows the performance of the advance routing strategy over all 12 DVRP instances with difference combinations of $ACPercent$ and $RequestProb$ values.

- When holding $ACPercent$ fixed, we identify significant increasing trends in all three measures as $RequestProb$ increases. As the number of dynamic customers increases, the advance routing strategy performs worse because it only considers advance customers in route construction and optimization. Dynamic customers are routed by using a myopic cheapest insertion heuristic only after they are realized.
- On the other hand, when holding $RequestProb$ fixed, we identify significant decreasing trends in all three measures as $ACPercent$ increases. This can be explained by the inverse of the argument above.
- We also notice that the advance routing strategy generates near-static behavior in cases corresponding to the highest value of $ACPercent$ tested, which is 75% (bottom row of the table). In these cases, advance customers make up the majority of all

customers, and the problem is only a small deviation away from a static vehicle routing problem. With the lack of dynamic customers, the advance routing strategy by design is similar to the static strategy, which assumes that all dynamic customers are known at the beginning of the day, and can be treated the same as advance customers.

Table 5.1 Simulation Results of Advance Routing Strategy

RequestProb	0.25			0.5			0.75		
ACPercent	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.1	24.78%	0.75	1.05	28.20%	1.65	2.45	35.00%	3.15	4.60
0.25	18.76%	0.65	0.80	22.00%	1.30	1.85	27.03%	2.35	3.30
0.5	7.02%	0.30	0.45	14.61%	1.20	1.55	15.08%	1.30	1.95
0.75	2.49%	0.00	0.10	6.56%	0.35	0.55	7.46%	0.10	0.30

5.3.2 Look-ahead Dynamic Routing

Table 5.2 shows the results of the look-ahead dynamic routing strategy on the base case scenario. The base case scenario is designed by setting both instance parameters to a moderate level, namely $ACPercent = 0.5$ and $RequestProb = 0.5$. Results of the advance routing strategy is included in the right-most column for reference.

- When holding the $forecastHorizon$ fixed, we see significant decreasing trends in all three measures as the $threshold$ increases. The higher the threshold, the fewer customers we anticipate, and thus the higher the quality of anticipation, because the chance that these anticipated customers will actually request service and become confirmed is higher. This observation identifies that we want to be selective when constructing the set of anticipated customers, because the threshold forecast procedure implicitly assumes that all of the anticipated customers will request service.
- When holding the $threshold$ fixed, the strategy with $forecastHorizon = 40$ always sees the best performance. $forecastHorizon = 40$ corresponds to looking ahead roughly 2 time periods. Since the look-ahead dynamic routing strategy periodically re-optimizes the current vehicle routes, it is sufficient to look ahead a relatively short horizon. When we look ahead for too long, the accuracy of the forecast drops. However, too short a forecast horizon will not generate enough

information to construct an insightful forecast, as shown by the results when $forecastHorizon = 20$.

- In the base case, the look-ahead routing strategy outperforms the advance routing strategy in terms of total travel distance. However, the advance routing strategy outperforms the look-ahead routing strategy in terms of vehicle usage. The reason behind this observation lies in the fact that the look-ahead dynamic routing strategy reserves space on vehicle routes for anticipated customers. These spaces are not freed until either the customer requests service, or the next decision epoch is reached. This design will cause infeasibility of requests made by non-anticipated customers. On the contrary, the advance routing strategy does not forecast any customer requests. By scheduling only confirmed customers, this strategy always offers more available space on the current vehicle routes as compared to the look-ahead routing strategy. Thus the advance routing strategy shows its merits in terms of maximizing vehicle utilization and minimizing the number of rejected customers.

Table 5.3 shows the results of the look-ahead dynamic routing strategy by varying the parameter $requestProb$ and holding $ACPercent$ at 50%. When the $requestProb$ is low at 25%, the setting of $forecastHorizon = 40$ generates solutions with the smallest distance penalty on average when the threshold is set to 0.2. However, a shorter forecast horizon of 20 time steps generates better solutions as measured by vehicle usage and the number of rejected customers. It shows that when the number of dynamic customers is small, it may be sufficient to forecast a short horizon, because there is not much information to explore before reaching the next decision epoch. As the $requestProb$ increases to 75%, the setting of $forecastHorizon = 40$ still generates solutions with the smallest distance penalty on average. However, a longer forecast horizon of 60 time steps generates better solutions as measured by vehicle usage and the number of rejected customers. Equivalently speaking, as more dynamic customers request service, the value of forecasting increases. Longer forecast horizons provide more information on dynamic customer requests, thus leading to better solutions in these cases. In short, as the value of $requestProb$ increases, the value of $forecastHorizon$ that corresponds to the best solutions also tends to increase, when holding $ACPercent$ at 50%.

Table 5.4 shows the result of the look-ahead dynamic routing strategy by varying the parameter $ACPercent$, and holding $requestProb$ at 50%. When the value of the $ACPercent$ is

below the base case value, at 10% and 25%, there are relatively less advance customers. In such cases, a shorter forecast horizon of 20 time steps outperforms the previous-identified best strategy with $forecastHorizon = 40$, especially in terms of vehicle usage and the number of rejected customers. This is because when the probability that a dynamic customer requests service is high, a long forecast horizon will include too many customers in the anticipated set. The probability that all anticipated customers request service drops significantly. Therefore the quality of the forecast drops. Anticipated customers hold up spaces on vehicle routes, increasing the chance that the request of a non-anticipated customer is rejected, as shown by poor vehicle usage and large numbers of rejected customers associated with long forecast horizons. When the $ACPercent$ increases above the base case value to 75%, we see fairly flat behavior of the look-ahead dynamic routing strategy across all parameter settings. In such cases, we are faced with a large number of advance customers, and very few dynamic customers. The problem is rather static and is insensitive to the length of forecast horizon and the level of threshold used by the forecast procedure.

Table 5.2 Look-ahead Dynamic Routing in Base Case

ACPercent = 0.5, requestProb = 0.5																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.100	8.29%	2.90	4.00	7.78%	3.15	5.38	9.02%	3.55	5.00	7.93%	3.50	5.80	8.41%	3.40	6.22			
0.250	7.08%	2.15	1.50	5.95%	2.95	4.40	6.68%	3.05	5.25	7.34%	3.50	6.00	6.92%	3.20	5.64	14.61%	1.20	1.55
0.400	7.05%	1.95	1.33	4.83%	1.90	1.57	6.03%	2.30	3.43	6.52%	3.05	5.15	6.65%	3.05	5.11			

Table 5.3 Look-ahead Dynamic Routing When Varying requestProb

ACPercent = 0.5, requestProb = 0.25																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.050	5.87%	2.90	4.50	7.55%	2.95	4.73	5.06%	2.75	4.75	8.82%	3.40	5.10	5.77%	3.05	4.00			
0.125	2.85%	1.85	2.33	4.13%	2.40	3.90	5.80%	2.85	4.60	6.13%	2.70	5.00	4.79%	2.50	4.20	7.02%	0.30	0.45
0.200	3.51%	1.65	1.00	2.78%	2.15	3.00	3.98%	2.25	3.00	4.38%	2.55	3.80	3.77%	2.35	3.00			
ACPercent = 0.5, requestProb = 0.5																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.100	8.29%	2.90	4.00	7.78%	3.15	5.38	9.02%	3.55	5.00	7.93%	3.50	5.80	8.41%	3.40	6.22			
0.250	7.08%	2.15	1.50	5.95%	2.95	4.40	6.68%	3.05	5.25	7.34%	3.50	6.00	6.92%	3.20	5.64	14.61%	1.20	1.55
0.400	7.05%	1.95	1.33	4.83%	1.90	1.57	6.03%	2.30	3.43	6.52%	3.05	5.15	6.65%	3.05	5.11			
ACPercent = 0.5, requestProb = 0.75																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.150	6.37%	2.65	3.33	4.74%	2.70	3.50	4.10%	2.70	3.17	4.04%	2.55	4.70	5.15%	2.55	4.10			
0.375	4.86%	1.90	1.88	5.54%	2.60	4.29	3.94%	2.60	3.38	5.26%	2.45	4.80	3.77%	2.30	3.50	15.08%	1.30	1.95
0.600	3.01%	2.25	1.33	2.65%	2.20	2.00	2.71%	1.80	2.00	3.31%	2.45	2.86	3.58%	2.45	3.44			

Table 5.4 Look-ahead Dynamic Routing When Varying ACPercent

ACPercent = 0.1, requestProb = 0.5																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.100	12.36%	3.60	7.68	13.20%	4.15	8.79	12.18%	4.45	10.28	10.43%	3.80	11.70	13.24%	4.20	11.20			
0.250	6.37%	2.10	1.88	9.83%	3.45	7.63	11.81%	3.95	10.00	10.19%	3.85	11.05	12.61%	4.10	10.83	28.20%	1.65	2.45
0.400	7.62%	1.50	2.00	5.52%	1.95	3.13	7.72%	2.75	5.92	10.27%	3.05	9.11	9.58%	3.30	9.00			
ACPercent = 0.25, requestProb = 0.5																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.100	9.88%	3.85	6.71	11.45%	3.90	8.29	11.55%	3.90	8.27	9.25%	3.90	8.71	10.91%	4.10	8.65			
0.250	6.05%	1.95	2.13	7.76%	3.25	6.82	9.01%	3.95	8.06	9.12%	3.40	8.22	11.27%	3.85	8.38	22.00%	1.30	1.85
0.400	5.84%	1.65	1.50	8.00%	2.40	3.50	7.37%	2.60	4.73	7.84%	2.95	6.81	8.45%	2.95	6.86			
ACPercent = 0.5, requestProb = 0.5																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.100	8.29%	2.90	4.00	7.78%	3.15	5.38	9.02%	3.55	5.00	7.93%	3.50	5.80	8.41%	3.40	6.22			
0.250	7.08%	2.15	1.50	5.95%	2.95	4.40	6.68%	3.05	5.25	7.34%	3.50	6.00	6.92%	3.20	5.64	14.61%	1.20	1.55
0.400	7.05%	1.95	1.33	4.83%	1.90	1.57	6.03%	2.30	3.43	6.52%	3.05	5.15	6.65%	3.05	5.11			
ACPercent = 0.75, requestProb = 0.5																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.100	5.06%	2.40	1.40	6.39%	2.45	2.00	6.49%	2.40	1.25	5.90%	2.00	2.25	4.58%	2.20	1.50			
0.250	6.54%	2.15	0.00	6.38%	2.30	2.00	5.07%	2.25	1.80	5.28%	2.40	2.00	6.98%	2.45	2.17	6.56%	0.35	0.55
0.400	6.19%	2.20	1.50	5.14%	2.05	2.00	4.06%	1.80	1.67	5.00%	2.10	2.25	4.79%	2.15	1.83			

5.3.3 The Least Dynamic Case

One of the most extreme cases we have tested corresponds to a problem with little dynamism. This is represented by setting $ACPercent$ to its highest level and $RequestProb$ to its lowest value. Table 5.5 shows the results of the look-ahead dynamic strategy and the advance routing strategy on the least dynamic case. In this case, the advance routing strategy clearly outperforms the look-ahead dynamic routing strategy on all performance measures. Therefore, when there are few dynamic customers, a straightforward advance routing strategy is sufficient to solve the problem.

5.3.4 Comparison of Forecast Procedures

All the experimental results presented above are based on the threshold forecast procedure. We have also implemented the sampling forecast procedure. We now compare their performance in the base case. It can be clearly seen in Table 5.6 that the threshold procedure outperforms the sampling procedure by a very large margin as measured by total travel distance. And almost always outperforms the sampling procedure in the other two measures. This is because the sampling forecast procedure only draws one sample of potential future requests. Given that the total number of unconfirmed customers may be large, each sample alone has a very small probability of being the true realization of the randomness. Thus the sampling procedure only explores a very small fraction of the probability space, leading to a worse solution. The performance of the sampling procedure could be improved with more sampling and this is one avenue to explore in future research.

Table 5.5 The Least Dynamic Case

ACPercent = 0.75, requestProb = 0.25																		
forecastHorizon	20			40			60			80			100			Advance		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.050	7.27%	2.30	2.00	6.60%	2.00	1.00	5.52%	1.85	0.33	5.16%	2.10	0.67	4.89%	2.25	2.00			
0.125	4.63%	1.60	1.50	4.58%	1.95	1.25	3.55%	1.60	3.33	5.20%	1.75	0.00	5.82%	2.15	1.00	2.49%	0.00	0.10
0.200	5.48%	1.95	1.50	5.27%	1.80	2.00	5.41%	1.65	0.86	5.20%	1.80	1.50	4.89%	1.95	2.00			

Table 5.6 Comparison of Forecast Procedures

Threshold Forecast Procedure															
forecastHorizon	20			40			60			80			100		
threshold	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
0.100	8.29%	2.90	4.00	7.78%	3.15	5.38	9.02%	3.55	5.00	7.93%	3.50	5.80	8.41%	3.40	6.22
0.250	7.08%	2.15	1.50	5.95%	2.95	4.40	6.68%	3.05	5.25	7.34%	3.50	6.00	6.92%	3.20	5.64
0.400	7.05%	1.95	1.33	4.83%	1.90	1.57	6.03%	2.30	3.43	6.52%	3.05	5.15	6.65%	3.05	5.11
Sampling Forecast Procedure															
forecastHorizon	20			40			60			80			100		
	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected	Dist %	Vehicle	Rejected
	15.43%	3.45	5.42	16.31%	3.20	5.33	15.05%	3.80	5.55	14.80%	3.35	5.15	14.64%	3.40	4.55

6. Implementation

This project addresses a dynamic vehicle routing problem with uncertainties in customer requests. A typical application of this project is in the highly competitive trucking industry, i.e., companies consolidating shipments from multiple suppliers and transporting them to a central depot. Trucking is the dominant form of transportation used in moving goods into and out of the Los Angeles region, and has the highest level of interaction with other social functions. Methodologies that can improve the efficiency of truck movements will dramatically improve the overall logistics network and reduce traffic congestion in urban areas. In particular, this research addresses the reduction of truck traffic by reducing total vehicle miles and number of vehicles used through better planning when faced with uncertainties.

The scheduling heuristics developed in this research are tested on problem-specific instances derived from the well-known Solomon (1987) vehicle routing problem instances. The performance of the proposed heuristics is compared with the performance of a static routing strategy and an advance routing strategy we develop. Meaningful insights can be drawn from the experimental results. The implementation of our heuristics will require suitable programming software tools such as C++, Java, etc. It also requires access to real-world customer request data such as distance and/or travel time between facilities, and historical information on the earliest and the latest time shipments are allowed to be picked up, and the service time of processing the shipments. The entire solution framework including all heuristics introduced is implemented in C++. The same code is used to generate the experimental results presented in the report.

7. Conclusions and Future Research

In this study, we consider a vehicle routing problem with dynamic customer requests. The problem is representative of the daily operation of many logistic service providers, especially those who consolidate shipments from multiple suppliers. We present a look-ahead vehicle routing framework based on request forecasts and periodic re-optimization of the current vehicle routes. The look-ahead feature consists of explicitly forecasting future customer requests based on stochastic information available on customer request times. We develop efficient construction, local search, and waiting time adjustment heuristics.

The look-ahead dynamic routing strategy we develop divides the planning horizon into multiple time periods of equal length. The beginning of each time period is called a decision epoch. At each decision epoch, a static vehicle routing problem is constructed by including both confirmed but not yet serviced customers at the current time and anticipated customers who are expected to request service based on the forecast. The static vehicle routing problem is solved by a construction heuristic. The initial solution is then improved by a simulated annealing-like local search heuristic, which balances speed and performance. Slack time along each vehicle route is adjusted dynamically and continuously during the day, by using two waiting time adjustment procedures, namely **Push Forward** and **Push Backward**. The behavior of the look-ahead dynamic routing strategy depends on multiple solution parameters, including the number of decision epochs, the length of look-ahead horizon, the forecast procedure, etc.

We explore experimentally the sensitivity of the look-ahead routing strategy on modified Solomon VRP instances. We use various settings of the instance parameters to reflect different real-world scenarios. For each scenario, we compare the quality of the solutions of the look-ahead dynamic routing strategy with the advance routing strategy. We are able to identify the best parameter settings of the look-ahead routing framework. In particular, when using the threshold forecast procedure to generate the set of anticipated customers, higher threshold values tend to out-perform lower threshold values. Based on our experimental results, the best-performing threshold values correspond to 0.75 times the probability that a dynamic customer requests service. In terms of the length of the forecast horizon, the best setting depends on the level of uncertainty of the problem. The range of the best value settings appears to be between 1 to 3 times the length of each time period. When the probability that a dynamic customer requests

service is higher, the length of the forecast horizon should be set longer, and vice versa. We see that the look-ahead routing strategy outperforms the advance routing strategy in terms of minimizing travel distance and vehicle usage in most of the scenarios, except when the problem is highly deterministic (with large number of advance customers and small number of dynamic customers).

One aspect not captured in the current design of the solution framework is the quality of forecast. Intuitively speaking, the longer the forecast horizon, the higher the level of uncertainties in the system, thus the lower the quality of forecasts. As forecasts become less accurate, it becomes less beneficial to apply the look-ahead routing strategy as proposed. Current version of the framework assumes that the probability distributions of conditional request times are perfectly known, while in fact they may only be known vaguely and the quality of our information drops for times further away from the current time.

References

Ang-Olson, J., & Ostria, S. (2005). Assessing the effects of freight movement on air quality at the national and regional level. Retrieved June 9, 2015. http://www.fhwa.dot.gov/environment/air_quality/publications/effects_of_freight_movement/chapter08.cfm

Bianchi, L. (2000). Notes on dynamic vehicle routing-the state of the art. Technical report, Istituto Dalle Molle di Studi sull'Intelligenza Artificiale.

Bent, R. W. &Van Hentenryck, P. (2007). Waiting and relocation strategies in online stochastic vehicle routing. *IJCAI*, 1816–1821.

Branke, J., Middendorf, M., Noeth, G., & Dessouky, M. M. (2005). Waiting strategies for dynamic vehicle routing. *Transportation Science*, 39(3), 298–312.

Chen, Z.-L., & Xu, H. (2006). Dynamic column generation for dynamic vehicle routing with time windows. *Transportation Science*, 40(1), 74–88.

Diana, M., & Dessouky, M. M. (2004). A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. *Transportation Research Part B: Methodological*, 38(6), 539-557.

Flatberg, T., Hasle, G., & Kloster, O. (2007). Dynamic and stochastic vehicle routing in practice. In *Dynamic Fleet Management*. 41–63.

Gendreau, M., Laporte, G., & Séguin, R. (1996). Stochastic vehicle routing. *European Journal of Operational Research*, 88(1), 3–12.

Laporte, G. (2009). Fifty Years of Vehicle Routing. *Transportation Science*, 43(4), 408–416.

Ioannou, G., Kritikos, M., & Prastacos, G. (2001). A greedy look-ahead heuristic for the vehicle routing problem with time windows. *Journal of the Operational Research Society*, 52(5), 523–537.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680.

LaHood, R., & Porcari, J. (2013). National transportation statistics. Bureau of Transportation Statistics. Retrieved June 9, 2015. http://www.rita.dot.gov/bts/sites/rita.dot.gov.bts/files/publications/national_transportation_statistics/index.html

Montemanni, R., Gambardella, L. M., Rizzoli, A. E., & Donati, A. V. (2005). Ant colony system for a dynamic vehicle routing problem. *Journal of Combinatorial Optimization*, 10(4), 327–343.

Montreuil, B. (2011). Toward a Physical Internet: meeting the global logistics sustainability grand challenge. *Logistics Research*, 3(2-3), 71–87.

Montreuil, B., Meller, R. D., & Ballot, E. (2013). Physical internet foundations. In T. Borangiu, A. Thomas, & D. Trentesaux (Eds.), *Service Orientation in Holonic and Multi Agent Manufacturing and Robotics* (Vol. 472, pp. 151–166). Berlin, Heidelberg: Springer Berlin Heidelberg.

Osman, I. H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, 41(4), 421–451.

Pillac, V., Gendreau, M., Gueret, C., & Medaglia, A.L., (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1), 1-11.

Potvin, J.-Y., & Rousseau, J.-M. (1995). An exchange heuristic for routing problems with time windows. *Journal of the Operational Research Society*, 46(12), 1433–1446.

Psaraftis, H. (1980). A dynamic-programming solution to the single vehicle many-to-many immediate request dial-a-ride problem, *Transportation Science*, 14(2), 130–154.
Savelsbergh, M. W. P. (1992). The vehicle routing problem with time windows: minimizing route duration. *INFORMS Journal on Computing*, 4(2), 146–154.

Secomandi, N., & Margot, F. (2009). Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operations Research*, 57(1), 214–230.

Solomon, M. M. (1987). Algorithms for the vehicle-routing and scheduling problems with time window constraints. *Operations Research*, 35(2), 254-265.

Sungur, I., Ordóñez, F., Dessouky, M. M., Ren, Y., & Zhong, H. (2010). A model and algorithm for the courier delivery problem with uncertainty. *Transportation Science*, 44(2), 193-205.

Teodorović, D., & Goran P. (1992). A simulated annealing technique approach to the vehicle routing problem in the case of stochastic demand. *Transportation Planning and Technology*, 16(4), 261-273.

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