Scenario-Based Planning for Partially Dynamic Vehicle Routing with Stochastic Customers

Russell W. Bent and Pascal Van Hentenryck

Department of Computer Science
Brown University
Providence, Rhode Island 02912

CS-02-08
July 2002

Scenario-Based Planning for Partially Dynamic Vehicle Routing with Stochastic Customers

Russell W. Bent and Pascal Van Hentenryck
Department of Computer Science
Brown University
Box 1910, Providence, RI 02912

July 24, 2002

Abstract

Multiple vehicle routing with time windows (VRPTW) is a hard and extensively studied combinatorial optimization problem. This paper considers a dynamic VRPTW with stochastic customers, where the goal is to maximize the number of serviced customers. It presents a multiple scenario approach (MSA) which continuously generates routing plans for scenarios including known and future requests. Decisions during execution uses a distinguished plan chosen, at each decision, by a consensus function. The approach was evaluated on vehicle routing problems adapted from the Solomon benchmarks with a degree of dynamism varying between 30% and 80%. They indicate that MSA exhibits dramatic improvements over approaches not exploiting stochastic information, that the use of consensus function improves the quality of the solutions significantly, and that the benefits of MSA increases with the (effective) degree of dynamism.

1 Introduction

The vehicle routing problem with time windows is a hard combinatorial optimization problem with many important applications in distribution and transportation. It has received considerable attention in the last decades and sophisticated local search procedures are now quite effective in finding near-optimal solutions in reasonable time. In recent years, increased attention has been devoted to dynamic and/or stochastic versions of the problem (e.g. [7, 9, 10, 11, 12, 16, 23, 26, 27, 28, 33]). The stochastic and dynamic versions are motivated by the inherent uncertainties arising in many industrial problems and technological developments such as onboard computers and communication systems, which give transportation systems the opportunity to update plans even after the vehicle has been deployed.

Almost all existing work, which we review later in the paper, focuses either on the stochastic version or on the dynamic version exclusively. In stochastic optimization, the goal is to find, given a VRPTW where some data is stochastic, an *a priori* routing plan which minimizes the expected objective function. These approaches often have a recourse function to correct the plan when constraints are violated. In dynamic optimization, customer requests are not known in advance and become available incrementally over time. Typically, optimization is performed on the known quantities until an event (such as a customer request) occurs. The plan is then adapted to accommodate the new request (if possible). Of course, in general, stochastic information is also available for the dynamic version of the VRTPW. How to exploit the stochastic information to meet as many customer requests as possible is the main issue addressed in the paper and it is often mentioned as a fundamental open research problem in this area (e.g., [6, 9, 14, 17]).

This paper considers the partially dynamic vehicle routing problem with time windows, where some customers are known at planning time, while others are dynamic. The goal is to service as many customers as possible given a fixed number of vehicles. Moreover, stochastic information is assumed to be available on the dynamic (or stochastic) customers, as is typically the case in practical applications. Note that partially dynamic vehicle routing problems may vary greatly in complexity according to the degree of dynamism, i.e., the ratio *dynamic customers/total customers* [21].

To tackle this dynamic stochastic vehicle routing problem, the paper proposes a multiple scenario approach (MSA). The key idea behind MSA is to continuously generate and solve scenarios which include both static and dynamic requests. Decisions during plan execution are based on a distinguished plan which also evolves over time. The distinguished plan is selected by a consensus function that selects the plan most similar to the current pool of routings. The consensus function implements a least-commitment approach, a well-known heuristic in the artificial intelligence community [31]. Experimental results on dynamic stochastic routing problems with a degree of dynamism between 30% and 80% indicate that MSA produces dramatic improvements over approaches not using stochastic information. They also indicate a strong synergy between MSA and the use of a consensus function, especially for problems with a high degree of dynamism or a large number of late dynamic customers (i.e., a high effective degree of dynamism in the terminology of [21]). It is also important to stress that the approach is essentially domain-independent and we expect that many other dynamic applications may benefit from these techniques, although our experimental results only consider our original VRPTW application.

The main contributions of this paper can be summarized as follows.

- The paper shows that MSA produces dramatic benefits in terms of solution quality by exploiting stochastic information on dynamic customers. This answers the open research issue mentioned in a variety of prior work (e.g., [6, 9, 14, 17]), at least for problems with a degree of dynamism between 30% and 80% (the moderately dynamic problems of [21]).
- The paper shows that it is beneficial to use the consensus function for choosing the

distinguished plan instead of travel distance, especially for problems with high degrees of dynamism. In fact, it is the synergy between stochastic information and the consensus function that seems to be responsible for the dramatic improvements over other approaches.

- The paper also shows that maintaining multiple routing plans is fundamental in obtaining high-quality solutions in dynamic routing problems, even when no stochastic information is available. This confirms and abstracts the results of [9] by making them independent of a specific local search.

The rest of the paper is organized as follows. Section 2 describes the problem formulation. Section 3 presents the traditional greedy approach. The main contributions of the paper are presented in the next three section. Section 4 introduces the MPA approach which continuously generates and maintains a pool of solutions but do not use stochastic information. Section 5 presents the MSA approach which generalizes MPA with stochastic information. Section 6 gives the experimental results. The last two sections present the related work and conclude the paper.

2 Problem Formulation

Customers Each problem contains N customer regions (regions for short) and N' customer service requests from the regions (customers for short). The customers are numbered $1 \ldots N'$, in the order that they make requests. The regions are numbered $1 \ldots N$. A depot is represented by the number 0. The set $\{0, 1, \ldots, N'\}$ represents all the sites in the problem. The travel cost between sites i and j is represented by c_{ij}. These travel costs satisfy the triangle inequality, i.e. $\forall_{i,j,k}(c_{ik} \leq c_{ij} + c_{jk})$. Every request has a demand $q_i \geq 0$ and a service time $s_i \geq 0$.

Vehicles Each problem has m identical vehicles available for use, each with capacity Q.

Routes A vehicle route, or route for short, starts at the depot, serves some number of customers at most once, and returns to the depot. Formally, a route is a sequence $[0, v_1, \ldots, v_n, 0]$, where $1 \leq v_i \leq N'$ and all v_i are distinct. The demand of a route is denoted by $q(r) = \sum_{i=1}^{n} q_i$. The travel cost of a route r, is denoted by $c(r)$ and is the cost of visiting all of its customers, i.e. $c(r) = c_{0v_1} + c_{v_1v_2} + \ldots + c_{v_{n-1}v_n} + c_{v_n0}$.

Routing Plan A routing plan, or plan for short, is a set of routes $\{r_1, \ldots, r_m\}$ servicing each customer exactly once. A routing plan assigns a unique successor and predecessor for each customer. For a plan, σ, the successor of customer i is denoted by $succ(i, \sigma)$ and the predecessor is denoted by $pred(i, \sigma)$. The travel cost of a plan is denoted by $c(\sigma)$, i.e. $c(\sigma) = \sum_{r=1}^{m} c(r)$. We also use $cust(r)$ and $cust(\sigma)$ to denote the customers of a route r and a plan σ.

Time Windows Each site i has a time window specified by an interval $[e_i, l_i]$, which represents the earliest and latest possible arrival times respectively. Vehicles must arrive by l_i, but can arrive before e_i. However, they are required to wait until e_i to begin service. Observe that e_0 represents the earliest time vehicles can leave the depot and l_0 represents the time all the vehicles must return to the depot.

A routing plan implicitly specifies the earliest service time a_i of each customer i, as well as the earliest arrival time $a(r)$ of each route r. These values are computed using simple recursive equations [18]. A plan also specifies the earliest and latest departure times from each customer and these quantities are used later in the paper.

A solution to the offline VRPTW is a routing plan $\sigma = \{r_1, \ldots, r_m\}$ that satisfies the capacity and time window constraints., i.e.,

$$\begin{cases} q(r_j) \leq Q & (1 \leq j \leq m) \\ a(r_j) \leq l_0 & (1 \leq j \leq m) \\ a_i \leq l_i & (\forall\ i\ \in cust(\sigma)) \end{cases}$$

The objective here is to find a routing plan maximizing the number of served customers $|cust(\sigma)|$.

In the dynamic VRPTW, customer requests are not known in advance and become available during the course of the day. In general, a number of requests are available initially, while others become available during plan execution. We assume that the distribution of the requests, or an approximation thereof, is available, which is typically the case in practical applications. For each incoming request, the dynamic algorithm must decide whether to accept or reject the customers. Once a request is accepted, it must be servived. (We also experimented with the simpler problem where requests were not necessarily serviced. The results were in fact rather similar in our limited experiments.)

3 The Greedy Algorithm

The greedy algorithm (e.g., [6, 14, 22, 21]) is the traditional approach to dynamic vehicle routing. It starts with a routing plan accommodating all initial requests and inserts new requests as they become available. The initial routing plan is, in general, obtained by a sophisticated local search, which is the only viable alternative for large-scale problems. When a request for customer c arrives, the greedy algorithm determines whether there is a feasible insertion point in the plan. If no such insertion point exists, the request is rejected. Otherwise, the new customer is inserted to minimize travel cost.

It is important to say a few words about plan execution. In general, a routing plan does not specify unique departure dates for customers but rather it constrains the earliest and the latest feasible departure times. In the dynamic setting, it is important to delay decisions as much as possible (since new requests may materialize), while not inducing any delay. Hence, the traditional strategy is to depart from a customer to arrive at the next customer as early as possible but not before the start of its time window. In other words, a vehicle never waits at a customer site before being serviced but may wait before departing to the next customer.

It is also interesting to observe that, in general, simple extensions to the greedy approach do not work well for vehicle routing with time windows. For instance, reoptimizing the plan after an insertion may often lead to solutions of poorer quality. The intuitive reason is that optimized solutions tends to produce tight schedules for the vehicles in use, which reduce their ability to accommodate future (late) customers. New, or less appropriate, vehicles must then be used for servicing these customers. More creative solutions must be used such as the adaptative memory approach of [9].

4 The Multiple Plan Approach

The Multiple Plan Approach (MPA) is a fundamental generalization of the greedy approach. Its key idea is to maintain a set of plans at every execution step. MPA was motivated by the seminal work of Gendreau et al. [9] which proposed a parallel tabu-search algorithm organized around multiple solutions and an adaptative memory. MPA abstracts and generalizes their approach by making it independent of the search procedure used to generate solutions. More precisely, MPA continuously generates routing plans that are consistent with the current decisions and deletes incompatible ones. Maintaining multiple plans is not sufficient in itself, since decisions must be taken regarding a specific plan to guarantee service of accepted requests. As a consequence, MPA maintains a distinguished plan at each execution step. The distinguished plan obviously evolves over time and all the plans maintained by MPA must be consistent with it. How to select the distinguished plan is a critical aspect of MPA, which we discuss later in the section.

More precisely, MPA handles four types of events: (1) customer requests, (2) vehicle departures, (3) plan generations, and (4) timeouts. Customer requests update the set of plans to accommodate the new request. Vehicle departures may render some routing plans invalid. The generation of a new plan may change the distinguished plan. Finally, some plans may become invalid over time. This arises when a plan specifies a vehicle departure (the latest departure time from a customer has been reached), while the distinguished plan specifies that the vehicle should wait at its current location. Timeouts capture these events. We now specify how to handle each event in more detail and discuss how to rank the plans next.

4.1 Event-Handling

At each time t, MPA maintains a set of plans S_t and a distinguished plan σ_t^*. For each event, we specify how to compute S_{t+1} and σ_{t+1}^* from past decisions, S_t, and σ_t^*. Each event type is specified in isolation, although several of them may occur simultaneously. It is easy to order them appropriately when this happens by selecting the events in the following order: timeouts, plan generations, customer requests, and vehicle departures. We make use of a set of functions f_t to rank the plans. Given a time t and a plan σ, $f_t(\sigma)$ returns a real value. Finally, observe that the implementation is event-driven. In other words, although we specify S_t and σ_t^* for all t, the implementation only considers the times where an actual event occurs.

Customer Request For a customer request c at time t, MPA must determine which plan in S_t can accommodate c. If none of them can, the request is rejected. Otherwise, the request is accepted and S_{t+1} is the set of plans where the requests has been inserted to minimize travel cost.

$F := \{\text{INSERT}(\sigma, c) \ \mid \ \sigma \in S_t \ \& \ \text{FEASIBLEINSERT}(\sigma, c)\};$
if $F \neq \emptyset$ then
    $S_{t+1} := F;$
    $\sigma_{t+1}^* := argmax(\sigma \in S_{t+1}) \ f_t(\sigma);$
else
    $S_{t+1} := S_t;$
    $\sigma_{t+1}^* := \sigma_t^*;$

where FEASIBLEINSERT(σ, c) returns true iff there is an insertion point in plan σ for customer c that satisfies the constraints, and where INSERT(σ, c) returns a plan σ' (if it exists) where c has been inserted in σ to minimize travel cost while satisfying the constraints.

Vehicle Departure When plan σ_t^* specifies that vehicle v must depart from customer c, it is necessary to remove all plans in S_t that are incompatible with this departure.

$S_{t+1} := \{\sigma \in S_t \mid \text{COMPATIBLE}(\sigma, \sigma_t^*, t)\};$
$\sigma_{t+1}^* := argmax(\sigma \in S_{t+1}) \ f_t(\sigma);$

where COMPATIBLE(σ, σ_t^*, t) is true if plan σ is compatible with plan σ_t^* up to time t. More formally, COMPATIBLE(σ, σ_t^*, t) holds iff

$$\forall c \in \text{DEPART}(t) : succ(\sigma_t^*, c) = succ(\sigma, c),$$

where DEPART(T) denotes the set of customers from which a vehicle departed before or at time t.

Plan Generation When a new plan σ is generated at time t, it is added to S_t and the new distinguished plan is recomputed. Note that plans are guaranteed to be compatible with the current distinguished plan, since they include all existing decisions and plan generation is cancelled whenever customer requests and vehicle departures occur.

$S_{t+1} := S_t \ \cup \ \{\sigma\};$
$\sigma_{t+1}^* := argmax(\sigma \in S_{t+1}) \ f_t(\sigma);$

Timeout At time t, some plan σ may become infeasible. This happens when a vehicle v is waiting at a customer c, while plan σ specifies that t is the latest departure for c.

$S_{t+1} := \{\sigma \in S_t \mid \text{FEASIBLE}(\sigma, t)\};$
$\sigma^*_{t+1} := \sigma^*_t;$

where $\text{FEASIBLE}(\sigma, t)$ holds if σ is feasible at time t:

$\forall c \in \text{DEPART}(t) : \text{LDT}(\sigma, succ(\sigma, c)) \leq t,$

where $\text{LDT}(\sigma, c)$ is the latest departure time from customer c in σ.

4.2 The Consensus Function

MPA is parametrized by the ranking functions f_t which selects the distinguished plan at each time t. An obvious choice for f_t would be to select the plan with the smallest travel cost. However, it is possible to do substantially better in practice, especially in instances where there are many late customers and/or many dynamic requests. The key idea is to use a consensus function which selects a plan most similar to other plans in S_t. Since the resulting distinguished plan does not depart from other plans too dramatically, consensus functions may be viewed as a least commitment strategy, a well-known heuristic in the artificial intelligence community [31]. More precisely, at each time t, the algorithm maintains a two-dimensional matrix M_t and $M_t[v, c]$ denotes the number of plans in S_t where vehicle v departs for customer c next. More formally, the matrix M_t is defined as

$$M_t[v, c] = \#\{\sigma \in S_t \mid succ(\sigma, \text{LDC}(v)) = c\},$$

where $\text{LDC}(v)$ is the last customer from which vehicle v departed in the plan execution. The consensus function f_t is then defined as

$$f_t(\sigma) = \sum_{v=1}^{m} M_t[v, succ(\sigma, \text{LDC}(v)].$$

5 The Multiple Scenario Approach

MPA with the consensus function produces dramatic improvements in quality compared to the greedy algorithm. This section introduces a multiple scenario approach (MSA) which significantly outperforms MPA by exploiting stochastic information.

The key idea behind MSA consists of generating new routing plans for scenarios that include both existing requests as well as possible future requests. Future requests are simply obtained by sampling their probability distributions. Once a routing σ is obtained for such a scenario, MSA projects σ on the known requests by removing future customers from the plan. The resulting *projected plan* σ^- leaves room for accommodating future requests if they materialize and is added to the pool S_t. MPA closely resembles MPA, since its basic event-handling procedures are essentially the same. Only plan generation differs in that it uses scenarios involving future requests (instead of known requests only). The solution to these scenarios are then projected on known requests before being inserted in S_t.

6 Experimental Results

This section presents the experimental results. We first describe how we adapted the traditional Solomon benchmarks to the dynamic setting. We then specify a few parameters for the algorithms. We then compare the various approaches and analyze the role of stochastic information and the consensus function in the results. We also give some interesting observations on the scenario survival rates.

6.1 The Experimental Setting

The dynamic problems are generated from the Solomon benchmarks [30]. Each customer in the Solomon instances becomes a customer region in our stochastic problems. Our instances are generated so that the expected number of customer requests n is 100 and the expected number of requests from a particular customer region is 1. As a consequence, the instances will preserve much of the structure of the Solomon instances, many of which are very challenging.

Request Arrivals To specify the arrival distribution of customer requests, the time horizon $H = l_0 - e_0$ is divided 4 time periods. Period 0 corresponds to known requests (requests available before e_0). Periods 1 to 3 can be thought as representing morning, early afternoon, and late afternoon. Each customer region is labeled with a time region according to its time window and its distance to the depot. The distance to the depot is taken into account to avoid generating requests which clearly cannot be serviced. There are no request labeled with period 3 (late afternoons). If region i is labeled with period 0, then this region requests service before e_0. If the region is labeled with period 1, there is a request for service before e_0 with probability 0.5 and a request for service in period 1 with probability 0.5. If the region is labeled with period 2, then there are four possibilities leading to four classes of instances. Problems in Class 1 have many initial requests and few late requests. Problems in Class 2 have many initial requests and many late requests. Problems in Class 3 have many initial requests and a mix of early and late requests. Finally, problems in Class 4 have few initial requests and a lot of late requests. More precisely, in Class 1 problems, there is a request before e_0 with probability 0.5, a request in period 1 with probability 0.4, and a request period 2 with probability 0.1. In Class 2 problems, there is a request before e_0 with probability 0.5, a request in period 1 with probability 0.1, and a request period 2 with probability 0.4. In Class 3 problems, a request is generated according to Class 1 with probability 0.5 and according to class 2 with probability 0.5. In Class 4 problems, there is a request before e_0 with probability 0.2, a request in period 1 with probability 0.2, and a request period 2 with probability 0.6. Observe that these probabilities are independent as to allow multiple or no requests from a particular region.

The instances were generated using these probabilities. If there is a request in time period k, then the request arrival time for customer c is drawn uniformly at random from the time interval $[(k-1) * \frac{H}{3}, \min(\lambda_c, k * \frac{H}{3} - 1)]$, where λ_c is the latest time a vehicle

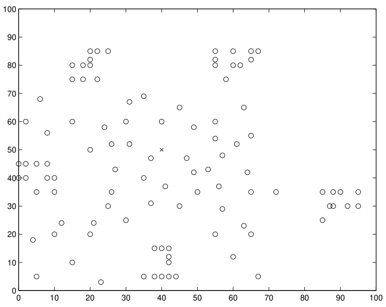

Figure 1: Customer Region Locations

can depart from the depot, service c, and return to the depot. The probabilities for each customer regions are available to a dynamic algorithm.

Instance Data The instances take their data (customer locations, time windows, demands, service times) from the class RC in the Solomon benchmarks. RC problems have a mix of clustered and randomly distributed customers, which makes them particularly interesting. Moreover, we selected problems with diverse time windows. In particular, RC101 problems have a good mix of short and long time windows and have a high percentage of 1-region customers. RC102 problems have many regions with short time windows, 45 1-region customers, and 45 2-region customers. RC104 problems have regions with extremely long time windows and 83 2-region customers. Figure 1 shows the location of the customer regions in Solomon's RC class. As a result, the instances cover a wide spectrum of possibilities and structures.

The number of vehicles available for the dynamic algorithms was determined by solving the offline problems (i.e., assuming that all requests are known in advance) and adding two vehicles. The offline problems were solved using the state-of-the-art vehicle routing algorithm (LS) presented in [2].

In the Solomon RC instances, the horizon is 240 time units, which is too short for the units to be seconds. In order to run a large number of experiments, the units cannot be minutes either. As a consequence, to obtain a realistic experimental setting, we scaled all spatial positions, time windows, travel costs, and service times by the factor $\frac{y}{H}$. By choosing $y = 30$ and minutes as units, we obtain meaningful instances which can be experimented with in reasonable time. Note that all results are reported in the original units for readers who are familiar with Solomon benchmarks.

Algorithms All algorithms were run on a Sun Ultra 10 with Sun's c++ compiler. The MPA and MSA algorithms generate 50 initial plans using the local search algorithm mentioned earlier. Each plan is optimized for 30 seconds. During plan execution, plans are generated by using the local search algorithm for 10 seconds.

Table 1: Class 1 Problem Results

	Greedy		MPA^d		MPA^c		MSA^d		MSA^c	
	C	V	C	V	C	V	C	V	C	V
rc101-1	4.0	16.0	1.6	15.8	1.8	16.0	1.0	16.0	**0.6**	**16.0**
rc101-2	10.0	16.0	3.2	16.0	3.0	16.0	**1.8**	**16.0**	2.6	16.0
rc101-3	9.0	15.0	1.6	15.0	1.2	15.0	1.8	15.0	**1.0**	**15.0**
rc101-4	3.0	17.0	0.2	16.8	**0.0**	17.0	**0.0**	**17.0**	0.2	17.0
rc101-5	5.0	17.0	2.0	17.0	1.4	17.0	**0.4**	**17.0**	1.0	17.0
Avg	6.20	16.2	1.72	16.1	1.48	16.2	**1.00**	**16.2**	1.08	16.2
rc102-1	6.0	14.0	3.0	14.0	2.2	14.0	**2.0**	**14.0**	2.4	14.0
rc102-2	8.0	13.0	1.0	13.0	2.4	13.0	**0.4**	**13.0**	0.8	13.0
rc102-3	11.0	15.0	2.4	15.0	4.8	15.0	1.0	15.0	**0.8**	**15.0**
rc102-4	8.0	14.0	1.8	13.8	1.8	14.0	**1.2**	**14.0**	1.4	14.0
rc102-5	5.4	15.0	1.8	15.0	2.2	15.0	1.2	15.0	**0.6**	**15.0**
Avg	7.68	14.2	2.00	14.2	2.68	14.2	**1.16**	**14.2**	1.20	14.2
rc104-1	5.0	11.0	1.4	11.0	0.2	11.0	**0.0**	**11.0**	0.2	11.0
rc104-2	15.6	12.0	0.8	12.0	1.0	12.0	**0.0**	**11.8**	**0.0**	12.0
rc104-3	17.2	13.0	**0.0**	12.6	**0.0**	12.8	**0.0**	**12.4**	**0.0**	13.0
rc104-4	12.2	12.0	**0.0**	**11.4**	0.4	12.0	**0.0**	11.8	0.2	12.0
rc104-5	7.6	11.0	**0.0**	11.0	0.8	11.0	**0.0**	**10.8**	**0.0**	11.0
Avg	11.52	11.8	0.44	11.6	0.48	11.8	**0.00**	**11.6**	0.04	11.8
Avg	8.47	14.07	1.39	14.0	1.55	14.1	**0.72**	**14.0**	0.77	14.1

6.2 The Results

Comparison of the Approaches Tables 1, 2, 3, and 4 compare the quality of the results produced by each approach. Time is not an issue here since vehicle travel is significantly larger than the computation time of our local seach algorithm. The tables report the performance of the greedy, MPA, and MSA approaches on each class of instances. A line in the tables report the average number of unserviced customers (C) and the number of used vehicles (V) in the final plan for 5 runs of the algorithms on the instance. We report the number of vehicles since some algorithms are able to reduce the number of vehicles and do not use all available vehicles. For MPA and MSA, we report results when travel distance (MPA^d and MSA^d) and the consensus function (MPA^c and MSA^c) are used as ranking functions.

The results can be informally summarized as follows.

1. MPA produces significant improvements over the greedy algorithm. In the average, MPA has about 1.5, 3.5, 2.3, and 6 unserviced customers for classes 1 to 4, while the greedy algorithm has about 8.5, 9, 9, and 16 unserviced customers. This confirms the results of Gendreau et al [9], since MPA^d is a generalization and abstraction of their work.

2. MSA produces significant improvements over MPA. In the average, MSA has about 0.75, 1.2, 1, and 2 unserviced customers for classes 1 to 4. The benefits are particularly clear on the rc104 instances from Class 4, where the number of unserviced customers

Table 2: Class 2 Problem Results

	Greedy		MPA^d		MPA^c		MSA^d		MSA^c	
	C	V	C	V	C	V	C	V	C	V
rc101-1	2.0	13.0	0.6	13.0	2.0	13.0	**0.0**	**13.0**	0.2	13.0
rc101-2	3.0	14.0	2.0	14.0	2.0	14.0	**1.0**	**14.0**	1.4	14.0
rc101-3	7.0	17.0	0.4	17.0	0.8	17.0	**0.0**	**17.0**	**0.0**	17.0
rc101-4	6.0	17.0	1.8	17.0	2.2	17.0	**0.6**	**17.0**	0.8	17.0
rc101-5	6.0	16.0	1.8	16.0	**1.4**	**16.0**	1.6	16.0	**1.4**	16.0
Avg	4.80	15.4	1.32	15.4	1.68	15.4	**0.64**	**15.4**	0.76	15.4
rc102-1	8.2	15.0	**0.0**	15.0	0.4	15.0	**0.0**	**14.6**	0.4	15.0
rc102-2	8.6	14.0	1.4	14.0	1.2	14.0	**0.6**	**14.0**	1.2	14.0
rc102-3	10.0	14.0	3.0	14.0	3.2	14.0	**2.0**	**14.0**	**2.0**	14.0
rc102-4	6.0	15.0	0.6	15.0	0.8	15.0	**0.2**	**15.0**	0.4	15.0
rc102-5	8.0	14.0	3.0	14.0	3.4	14.0	**2.6**	**14.0**	2.8	14.0
Avg	8.16	14.4	1.60	14.4	1.80	14.4	**1.08**	**14.3**	1.36	14.4
rc104-1	15.2	12.0	11.2	12.0	8.6	12.0	6.2	12.0	**3.0**	**12.0**
rc104-2	17.8	12.0	9.4	12.0	10.0	12.0	5.4	12.0	**2.6**	**12.0**
rc104-3	16.4	12.0	7.8	12.0	7.2	12.0	2.0	12.0	**0.8**	**12.0**
rc104-4	11.0	13.0	2.4	13.0	2.0	13.0	0.8	13.0	**0.6**	**13.0**
rc104-5	14.6	12.0	5.2	12.0	9.0	12.0	4.2	12.0	**0.2**	**12.0**
Avg	15.00	12.2	7.20	12.2	7.36	12.2	3.72	12.2	**1.44**	**12.2**
Avg	9.32	14.0	3.37	14.0	3.61	14.0	1.81	14.0	**1.19**	**14.0**

Table 3: Class 3 Problem Results

	Greedy		MPA^d		MPA^c		MSA^d		MSA^c	
	C	V	C	V	C	V	C	V	C	V
rc101-1	8.0	15.0	1.6	15.0	1.2	15.0	**0.6**	**15.0**	0.8	15.0
rc101-2	8.8	16.0	1.4	16.0	**1.2**	**16.0**	2.2	16.0	1.4	16.0
rc101-3	6.0	14.0	1.0	14.0	1.0	14.0	**0.6**	**13.8**	0.8	14.0
rc101-4	7.6	17.0	1.4	17.0	2.0	17.0	**0.6**	**17.0**	1.0	17.0
rc101-5	4.0	16.0	0.6	16.0	1.2	16.0	**0.0**	**15.4**	0.8	16.0
Avg	6.88	15.6	1.20	15.6	1.32	15.6	**0.80**	**15.6**	0.96	15.6
rc102-1	6.0	15.0	1.8	15.0	2.2	15.0	**1.6**	**15.0**	**1.6**	15.0
rc102-2	5.0	14.0	2.4	14.0	3.8	14.0	**0.8**	**14.0**	1.8	14.0
rc102-3	9.0	13.0	2.0	13.0	1.6	13.0	**0.8**	**13.0**	**0.8**	13.0
rc102-4	6.0	15.0	3.2	15.0	2.6	15.0	**0.8**	**15.0**	1.8	15.0
rc102-5	7.0	15.0	1.2	15.0	2.6	15.0	**1.4**	**15.0**	1.6	15.0
Avg	6.60	14.4	2.12	14.4	2.56	14.4	**1.08**	**14.4**	1.52	14.4
rc104-1	20.2	12.0	8.2	12.0	5.6	12.0	4.8	12.0	**2.4**	**12.0**
rc104-2	9.0	12.0	1.6	12.0	1.0	12.0	1.0	12.0	**0.2**	**12.0**
rc104-3	12.0	12.0	2.8	12.0	1.8	12.0	1.4	12.0	**0.4**	**12.0**
rc104-4	12.0	12.0	2.4	12.0	2.2	12.0	1.6	12.0	**0.2**	**12.0**
rc104-5	17.4	12.0	2.8	12.0	4.4	12.0	2.2	12.0	**0.6**	**12.0**
Avg	14.12	12.0	3.56	12.0	3.00	17.0	2.20	17.0	**0.76**	**12.0**
Avg	9.20	14.0	2.29	14.0	2.29	14.0	1.36	14.0	**1.08**	**14.0**

Table 4: Class 4 Problem Results

	Greedy		MPA^d		MPA^c		MSA^d		MSA^c	
	C	V	C	V	C	V	C	V	C	V
rc101-1	3.0	16.0	**0.0**	**15.8**	1.0	16.0	**0.0**	**16.0**	1.0	16.0
rc101-2	8.0	15.0	**2.2**	**15.0**	3.4	15.0	2.8	15.0	3.6	15.0
rc101-3	4.0	16.0	0.6	16.0	1.6	16.0	**0.0**	**16.0**	1.6	16.0
rc101-4	11.0	17.0	**1.0**	**17.0**	1.2	17.0	**1.0**	17.0	1.4	17.0
rc101-5	11.8	16.0	2.8	16.0	3.2	16.0	**2.0**	**16.0**	2.2	16.0
Avg	7.56	16.0	1.32	16.0	2.08	16.0	**1.16**	**16.0**	1.96	16.0
rc102-1	6.0	15.0	0.2	14.2	**0.0**	15.0	**0.0**	**14.8**	0.4	15.0
rc102-2	9.0	15.0	1.8	15.0	**1.4**	15.0	1.6	15.0	**1.4**	**15.0**
rc102-3	15.6	15.0	1.8	15.0	1.0	15.0	**0.2**	**15.0**	1.4	15.0
rc102-4	3.6	14.0	0.4	13.8	**0.0**	14.0	**0.0**	**14.0**	**0.0**	14.0
rc102-5	5.0	15.0	**0.2**	**15.0**	1.4	15.0	0.6	15.0	0.6	15.0
Avg	7.84	14.8	0.88	14.6	0.76	14.8	**0.48**	**14.8**	0.68	14.8
rc104-1	41.0	13.0	16.8	13.0	15.0	13.0	15.6	13.0	**3.2**	**13.0**
rc104-2	41.0	14.0	18.2	14.0	15.8	14.0	16.0	14.0	**3.4**	**14.0**
rc104-3	26.0	13.0	16.4	13.0	15.2	13.0	13.8	13.0	**5.6**	**13.0**
rc104-4	23.0	12.0	16.0	12.0	18.2	12.0	15.6	12.0	**2.4**	**12.0**
rc104-5	27.8	11.0	14.4	11.0	11.0	11.0	8.2	11.0	**2.0**	**11.0**
Avg	31.76	12.6	16.36	12.6	15.04	12.6	13.84	12.6	**3.32**	**12.6**
Avg	15.72	14.5	6.19	14.5	5.96	14.5	5.16	14.5	**1.99**	**14.5**

drops from about 15 to about 3.5 in the average. More generally, the improvements of MSA over MPA increase with the degree of (effective) dynamism.[1]

3. The consensus function brings substantial benefits over travel distance for high degree of (effective) dynamism and seems very robust in general. It is slightly outperformed by total distance for low degree of dynamism in general. It produces significant improvements for high degree of dynamism. In fact, *it is the synergy between stochastic information and the consensus function that seems to be responsible for the dramatic improvements over other approaches.*

The Benefits of Stochastic Information We now study the benefits of stochastic information in more detail. Figures 2 and 3 compare the quality of the solutions of greedy, MPA, and MSA approaches as a function of the degree of dynamism and as a function of the percentage late stochastic customers, which is related to the effective degree of dynamism. The results indicate that the benefits of MSA are particular dramatic for large percentages of stochastic customers and/or large percentages of late stochastic customers. Both the MPA and the MSA approaches use the consensus function here.

The Benefits of Consensus We now try to understand the impact of the ranking function on the quality of the results. Figures 4 and 5 compare the quality of the MSA algorithm

[1]Note that rc104 instances have many customers with late time windows and with large time windows, two features that seem to magnify the benefits of MSA.

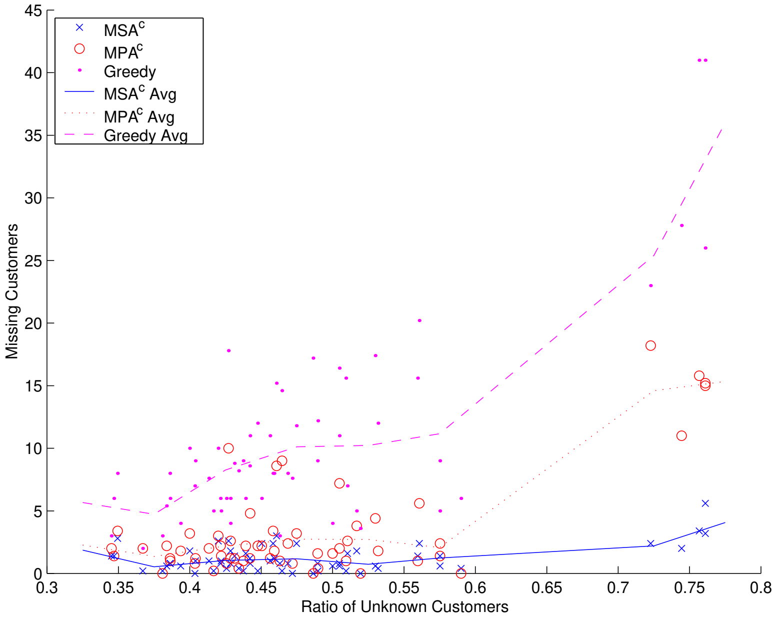

Figure 2: Quality wrt the Degree of Dynamism.

with two ranking functions: travel distance and the consensus function. The experimental results clearly show the value of the consensus function, especially when the degree of dynamism and/or the percentage of late stochatic customers increase. *As a consequence, the experimental results indicate that both stochastic information and the consensus function are critical in obtaining good quality solutions in these dynamic vehicle routing instances.* The use of a consensus function makes the MSA approach much more robust and allows it to accommodate many more (late) stochastic customers.

As a consequence, the synergy between MSA and the consensus function allows MSA to scale up to higher degree of dynamism, an open issue mentioned in [21].

Survival Rates It is interesting to understand how many plans are kept by MSA and the survival time of these plans. Figure 6 shows the size of S over time for a Class 3 problem. During plan execution, 244 distinct plans were generated. Observe the large drops in the size of S at time 0, 2, 5, 10, and 20. These drops were preceded by vehicle departure events, indicating that vehicle departures are the most likely events to trigger large reductions in S. Figure 7 shows the average, minimum, and maximum survival times of scenarios generated during 1 minute intervals for the same Class 3 problem. The -1 block represents the scenarios generated before e_0. From both figures, it is clear that the one scenario to survive until the end (at time 21, when the size of S dropped to 1) was created at time 19, as indicated by the large spike in the plot of the maximum. The general trend of survival

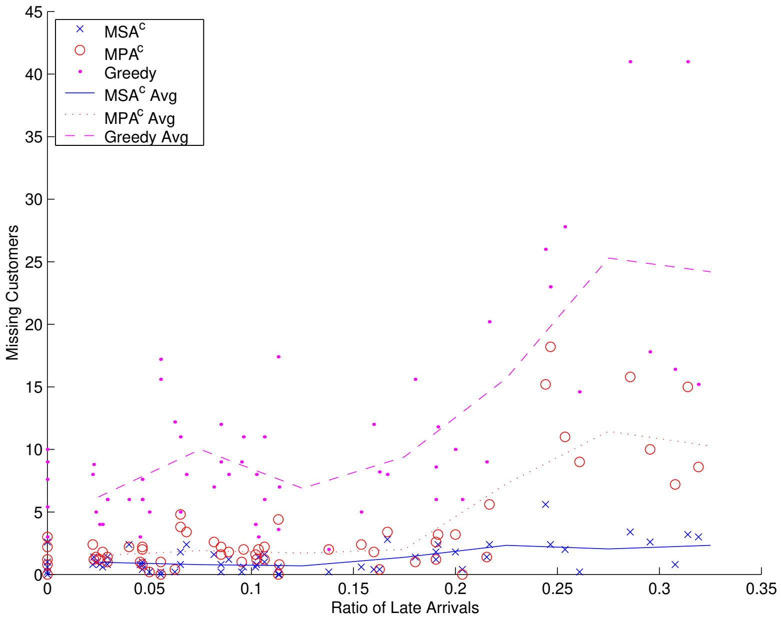

Figure 3: Quality wrt % Late Stochastic Customers.

time of the scenarios is to increase over time. This indicates that there is less variability in the possible plans as time goes on, which also points to how important early decisions are. This is more evidence that stochastic information is of clear benefit for dynamic vehicle routing.

7 Related Work

There are very few papers where stochastic information is used to solve dynamic vehicle routing problems. In general, papers focus either on static stochastic problems or on dynamic problems.

Most research on stochastic vehicle routing minimizes the expected travel distance. Generally, a simple recourse function (i.e., returning to the depot) is available during execution when feasibility constraints are violated. The recourse function adds some cost to the expectation. See [3, 19] for some early work and [12] for an overview of the various models and approaches. More recently, [33] studied the standard stochastic demand problem and adds the ability to preemptively return to the depot to unload capacity if the expectation makes it a better choice then a likely forced return later. [10] approaches the problem with stochastic demand and stochastic customers and solves the expectation optimally using linear programming techniques. [11] uses the same model but uses tabu search to solve larger instances of the problem. The recourse function in these problems is to return to the depot for unloading whenever capacity is violated. [23] treats the problem as a space

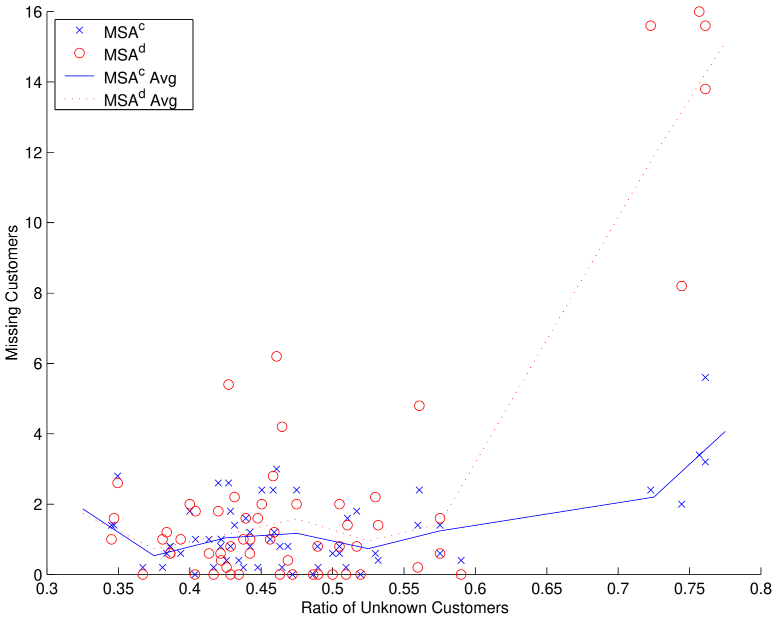

Figure 4: Quality wrt the Degree of Dynamism.

partitioning problem. Each vehicle is assigned regions of space and the goal is to create the regions such that the probability of failing to service all customers in the regions is below some threshold. [28] formulates the problem as a stochastic dynamic program and uses a reinforcement learning to approximate good solutions.

Most research in dynamic vehicle routing ignores stochastic information. See [26] for a survey of dynamic vehicle routing. The most relevant work is probably the parallel tabu-search algorithm of [9]. This algorithm is based on an adaptative memory which stores potential solutions. As new customer requests arrive, the algorithm uses those solutions that allow service of the customer. The algorithm may violate time windows but there is a penalty incorporated into the objective function for late arrivals. As a consequence, feasibility is greatly simplified, but the objective function is more complex. The algorithm does not exploit stochastic information and uses travel time to choose between different solutions. Thanks to the use of the adaptative memory and the maintenance of multiple solutions, the algorithm significantly outperforms the greedy approach on moderately dynamic problems. It motivated the design and implementation of our MPA approach, which can be seen as a generalization and abstraction of this algorithm by isolating the component for searching solutions and the ranking function. See also [8] which applies similar ideas to a dynamic pick and delivery problem and which suggests adding historical/stochastic information as an avenue of future research. The other most relevant work is the excellent paper [21], which defines the notion of "degree of dynamism", i.e., percentage of dynamic customers, and the notion of "effective degree of dynamism", which captures the lateness

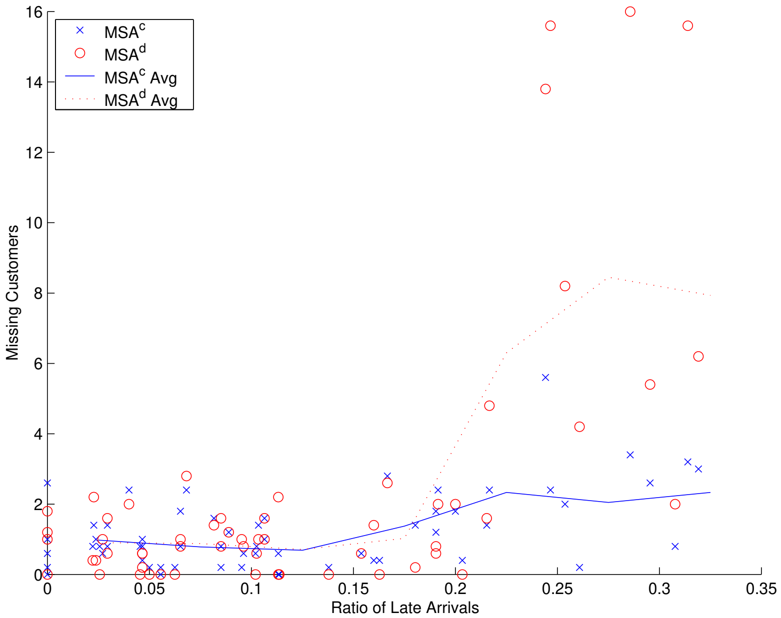

Figure 5: Quality wrt % Late Stochastic Customers.

of the dynamic customers, as well as its generalization to time windows (See also [20]). Our experimental setting is precisely based on these critical concepts. The paper also studies a dynamic problem where service time is an independent random variable. It compares several heuristics: first come/first serve, stochastic queue median, nearest neighbor, partitioning. The nearest neighbor heuristic, which is essentially our greedy heuristic, performs the best. Observe that the paper also indicates that little work has been done on incorporating stochastic information in a dynamic setting. References [4, 5] report some of the earliest work in mathematically studying some models of dynamic vehicle routing. They look a model with Poisson arrival rates for customers that are uniformly distributed in the plane. The goal is find a policy that minimizes the total system time of an infinite horizon (where system time is the total, for each customer, time spent servicing a customer and the wait time of a customer). The policies developed here (and a finite time model version of this) are assessed in [21]. Reference [32] uses the same model as [4, 5] and apply similar techniques and analysis to the pick-up and delivery problem. Reference [13] presents an algorithm for ambulance relocation, which consists of two subproblems. The first subproblem is to send an ambulance when a call comes in and the second problem is to relocate ambulances such that areas of the region are covered by at least two ambulances. The objective function is to minimize the region that is not double-covered and the cost of relocation to double-cover more of the region. [16] explores the benefits of allowing vehicles to change their destinations (diversion) when traveling. [27] uses integer programming techniques to optimize the current state of information, while [7] decomposes every entity of the problem

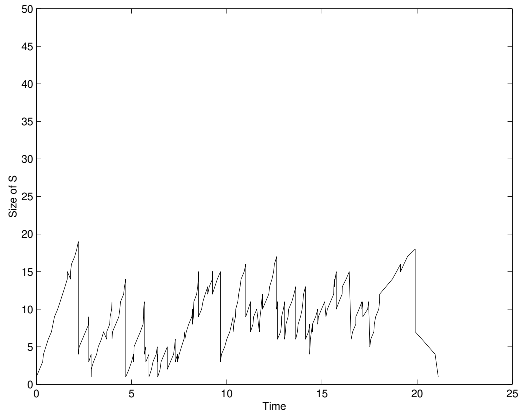

Figure 6: Size of S for Class 3 Problem RC104-3

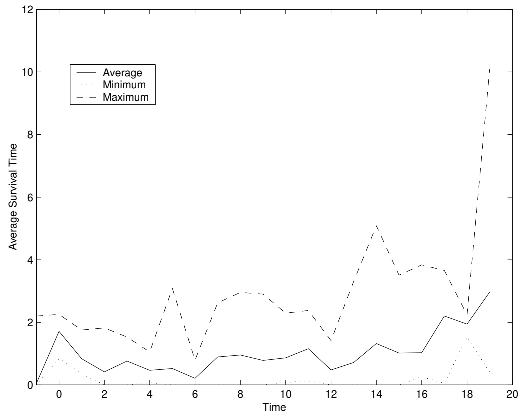

Figure 7: Average Survival Time for RC104-3

(i.e., vehicles, customers) into interactive agents working together to satisfy the current state of information. [34] is another agent approach to react to traffic jams. There have also been some work in proving competitive-ratio bounds for various versions of the VRP. These include [1] for finding an optimal 2-competitive algorithm for the online single vehicle dial-a-ride problem that minimizes the completion time. Other work on this problem has been performed by [15].

As mentioned, very little work uses stochastic information for dynamic vehicle routing. A notable exception is [29] which considers stochastic demands and only one vehicle. All customers are known and there are no time windows. They apply a rollout algorithm to approximate good solutions every time new information becomes available.

Finally, observe that, in the related field of resource allocation and assignment problems, there has been some recent work to incorporate stochastic and forecasted information into dynamic models (e.g., [24, 25]). Their work and our research both confirm the usefulness and applicablity of utilizing stochastic information.

8 Conclusion

This paper proposed a multiple scenario approach (MSA) to dynamic stochastic vehicle routing with time windows. MSA continuously solves scenarios which include both existing and future requests. Decisions use a distinguished plan selected by a consensus function. Experimental results indicate that MSA produces dramatic improvements over approaches not using stochastic information. They also indicate a strong synergy between MSA and the consensus function, especially for problems with many stochastic customers.

There are many research avenues to explore in this area. An intriguing question, which arises when demand exceeds capacity, is to develop strategies to reject customers if such a decision is likely to improve the quality of the plan. Similarly, generalizing the algorithm for additional stochastic data (e.g., demands, service times) is important in practice. Finally, as mentioned, the solutions presented in this paper are essentially domain-independent and we expect that many other dynamic applications may benefit from them.

Acknowledgments

Thanks to Eli Upfal for his help in designing a meaningful experimental setting. Russell Bent is supported by NDSEG Fellowship from the American Society of Engineering Education. Pascal Van Hentenryck is partially supported by NSF ITR Award DMI-0121495.

References

[1] N. Ascheuer, S. Krumke, and J. Rambau. Online Dial-A-Ride Problems: Minimizing the Completion Time. *STACS: 17th Annual Symposium on Theoretical Aspect of Computer Science*, pages 639–650, 2000.

[2] R. Bent and P. Van Hentenryck. A Two-Stage Hybrid Local Search for the Vehicle Routing Problem with Time Windows. *Technical Report, CS-01-06, Department of Computer Science, Brown University*, 2001.

[3] D. Bertsimas. A Vehicle Routing Problem with Stochastic Demand. *Operations Research*, 40:574–585, 1992.

[4] D. Bertsimas and G. Van Ryzin. A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane. *Operations Research*, 39:601–615, 1991.

[5] D. Bertsimas and G. Van Ryzin. Stochastic and Dynamic Vehicle Routing in the Euclidean Plane with Multiple Capacitated Vehicles. *Operations Research*, 41:60–76, 1993.

[6] L. Bianchi. Notes of Dynamic Vehicle Routing - The State of the Art. *Technical Report IDSIA-05-01*, 2000.

[7] H. Burckert, K. Fischer, and G. Vierke. Holonic Transport Scehduling with Teletruck. *Applied Artificial Intelligence*, 14:697–725, 2000.

[8] M. Gendreau, F. Guertin, J. Potvin, and R. Seguin. Neighborhood Search Heuristics for a Dynamic Vehicle Dispatching Problem with Pick-ups and Deliveries. *Tech. Rep. CRT-98-10, Centre de Recherche sur les Transport, Universite de Montreal*, 1998.

[9] M. Gendreau, F. Guertin, J. Y. Potvin, and E. Taillard. Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching. *Transportation Science*, 33 (4):381–390, 1999.

[10] M. Gendreau, G. Laporte, and R. Seguin. An Exact Algorithm for the Vehicle Routing Problem with Stochastic Demands and Customers. *Transportation Science*, 29:143–155, 1995.

[11] M. Gendreau, G. Laporte, and R. Seguin. A Tabu Search Heuristic for the Vehicle Routing Problem with Stochastic Demands and Customers. *Operations Research*, 44:469–477, 1996.

[12] M. Gendreau, G. Laporte, and R. Seguin. Stochastic Vehicle Routing. *European Journal of Operational Research*, 88:3–12, 1996.

[13] M. Gendreau, G. Laporte, and F. Semet. A Dynamic Model and Parallel Tabu Search Heuristic for Real-Time Ambulance Relocation. *Parallel Computing*, 27:1641–1653, 2001.

[14] M. Gendreau and J. Potvin. Dynamic Vehicle Routing and Dispatching. *Tech. Rep. CRT 97-38, Centre de Recherche sur les Transport, Universite de Montreal*, 1997.

[15] D. Hauptmeier, S. Krumke, and J. Rambau. The Online Dial-A-Ride Problem Under Reasonable Load. *Lecture Notes in Computer Science*, 1767:125–136, 2000.

[16] S. Ichoua, M. Gendreau, and J. Y. Potvin. Diversion Issues in Real-Time Vehicle Dispatching. *Transporation Science*, 34:426–438, 2000.

[17] P. Kilby, P. Prosser, and P. Shaw. Dynamic VRPs: A Study of Scenarios. *Technical Report APES-06-1998*, 1999.

[18] G. Kindervater and M. Savelsbergh. Vehicle Routing: Handling Edge Exchanges. In E. Aarts and J.K. Lenstra, editors, *Local Search in Combinatorial Optimization*, chapter 10, pages 337–360. John Wiley & Sons Ltd., 1997.

[19] G. Laporte, F. Louveaux, and H. Mercure. The Vehicle Routing Problem with Stochastic Travel Times. *Transportation Science*, 26:161–170, 1992.

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

[21] A. Larsen, O. Madsen, and M. Solomon. Partially Dynamic Vehicle Routing - Models and Algorithms. *Journal of the Operational Research Society*, 53:638–646, 2002.

[22] K. Lund, O. Madsen, and J. Rygaard. Vehicle Routing Problems With Varying Degress of Dynamism. *Technical Report, IMM, The Department of Mathematical Modeling, Technical University of Denmark*, 1996.

[23] A. Novaes and O. Graciolli. Designing Multiple-Vehicle Delivery Tours in a Grid-Cell Format. *European Journal of Operational Research*, 119:613–634, 1999.

[24] W. Powell. A Stochastic Formulation of the Dynamic Assignment Problem, with an Application to Truckload Motor Carriers. *Transportation Science*, pages 195–219, 1996.

[25] W. Powell, J. Shapiro, and H. Simao. A Representational paradigm for Dynamic Resource Transformation Problems. *Annals of Operations Reseach*, 104:231–279, 2001.

[26] H. Psaraftis. Dynamic Vehicle Routing: Status and Prospects. *Annals of Operations Research*, 61:143–164, 1995.

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

[28] N. Secomandi. Comparing Neuro-Dynamic Programming Algorithms for the Vehicle Routing Problem with Stochastic Demands. *Computers and Operations Research*, 27:1201–1225, 2000.

[29] N. Secomandi. A Rollout Policy for the Vehicle Routing Problem with Stochastic Demands. *Operations Research*, 49:796–802, 2001.

[30] M. M. Solomon. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35 (2):254–265, 1987.

[31] M. Stefik. Planning with Constraints (MOLGEN: Part 1). *Artificial Intelligence*, 16:111–139, 1981.

[32] M. Swihart and J. Papastavrou. A Stochastic and Dynamic Model for the Single-Vehicle Pick-up and Delivery Problem. *European Journal of Operational Research*, 114:447–464, 1999.

[33] W. Yang, K. Mathur, and R. Ballou. Stochastic Vehicle Routing Problem with Restocking. *Transportation Science*, 34:99–112, 2000.

[34] K. Zhu and K. Ong. A Reactive Method for Real Time Dynamic Vehicle Routing Problem. *12th ICTAI*, 2000.