Available online at www.sciencedirect.com
SCIENCE DIRECT®
European Journal of Operational Research 151 (2003) 1–11
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/dsw

Invited Review

Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies

Gianpaolo Ghiani [a], Francesca Guerriero [b], Gilbert Laporte [c,*], Roberto Musmanno [b]

[a] *Dipartimento di Ingegneria dell'Innovazione, Università degli Studi di Lecce, 73100 Lecce, Italy*
[b] *Dipartimento di Elettronica, Informatica e Sistemistica, Università degli Studi della Calabria, 87030 Rende (CS), Italy*
[c] *GERAD and Canada Research Chair in Distribution Management, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Montréal, Quebec, Canada H3T 2A7*

Received 3 June 2002; accepted 19 November 2002

Abstract

In real-time fleet management, vehicle routes are built in an on-going fashion as vehicle locations, travel times and customer requests are revealed over the planning horizon. To deal with such problems, a new generation of fast on-line algorithms capable of taking into account uncertainty is required. Although several articles on this topic have been published, the literature on real-time vehicle routing is still disorganized. In this paper the research in this field is reviewed and some issues that have not received attention so far are highlighted. A particular emphasis is put on parallel computing strategies.

Keywords: Parallel computing; Metaheuristics; Routing

1. Introduction

Vehicle routing problems (VRPs) are central to logistics management both in the private and public sectors. They consist of determining optimal vehicle routes through a set of users, subject to side constraints. The most common operational constraints impose that the total demand carried by a vehicle at any time does not exceed a given capacity, the total duration of any route is not greater than a prescribed bound, and service time windows set by customers are respected. In long-haul routing, vehicles are typically assigned one task at a time while in short-haul routing, tasks are of short duration (much shorter than a work shift) and a tour is to be built through a sequence of tasks. For a survey on the most relevant modeling and algorithmic issues on VRPs see the recent book by Toth and Vigo [1].

There exist several important problems that must be solved in real-time. In what follows, we

* Corresponding author. Tel.: +1-514-343-6143; fax: +1-514-343-7121.
E-mail addresses: gianpaolo.ghiani@unile.it (G. Ghiani), guerriero@deis.unical.it (F. Guerriero), gilbert@crt.umontreal.ca (G. Laporte), musmanno@unical.it (R. Musmanno).

0377-2217/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0377-2217(02)00915-3

review the main applications that motivate the research in the field of the real-time VRPs.

(i) *Dynamic fleet management:* Several large-scale trucking operations require real-time dispatching of vehicles for the purpose of collecting or delivering shipments. Important savings can be achieved by optimizing these operations [2–8].
(ii) *Vendor-managed distribution systems:* In vendor-managed systems, distribution companies estimate customer inventory level in such a way to replenish them before they run out of stock. Hence, demands are known beforehand in principle and all customers are *static*. However, because demand is uncertain, some customers (usually a small percentage) may run out of stock and have to be serviced urgently [9,10].
(iii) *Couriers:* Long-distance courier need to collect locally outbound parcels before sending them to a remote terminal to consolidate loads. Also, loads coming from remote terminals have to be distributed locally. Most pick-up requests are dynamic and have to be serviced the same day if possible [11–13].
(iv) *Rescue and repair service companies:* There are several companies providing rescue or repair services (broken car rescue, appliance repair, etc.) [14–16].
(v) *Dial-a-ride systems:* Dial-a-ride systems provide transportation services to people between given origin–destination pairs. Customers can book a trip one day in advance (*static customers*) [17] or make a request at short notice (*dynamic customers*) [18,19].
(vi) *Emergency services:* Emergency services comprise police, fire fighting and ambulance services. By definition, all customers are dynamic. Moreover, the demand rate is usually low so that vehicles become idle from time to time. In this context, relocating idle vehicles in order to anticipate future demands or to escape from downtown rush hour traffic jam is a major issue [20–22].
(vii) *Taxi cab services:* In taxi cab services, almost every customer is dynamic. As in emergency services, relocating temporary idle vehicles is an issue.

Due to recent advances in information and communication technologies, vehicle fleets can now be managed in real-time. When jointly used, devices like geographic information systems (GIS), global positioning systems (GPS), traffic flow sensors and cellular telephones are able to provide relevant real-time data, such as current vehicle locations, new customer requests and periodic estimates of road travel times [16]. If suitably processed, this large amount of data can be in principle be used to reduce cost and improve service level. To this end, revised routes have to be timely generated as soon as new events occur.

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

The aim of this article is to review some of the existing literature on local area real-time routing and dispatching problems with an emphasis on potential developments in the field of parallel computing. Our purpose is not to provide an extensive coverage of all existing algorithms. Rather we wish to describe the key concepts and trends in the area of real-time routing in the hope of stimulating further research.

The remainder of this article is organized as follows. In Section 2 the main features of real-time VRPs are illustrated, while in Sections 3 and 4 some of the most important algorithms are reviewed. Finally, in Section 5 a number of unexplored real-time VRPs are identified and new solution approaches are proposed, including several parallelization strategies for speeding up route re-optimization.

2. Definitions and concepts

In this section, we provide a simple classification of real-time VRPs and we examine some of their main features.

2.1. Problem classification

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

(i) *Static VRPs:* A static problem can be either deterministic or stochastic. In deterministic and static VRPs (like the classical capacitated VRP surveyed in [1]) all data are known in advance and time is not taken into account explicitly. In stochastic and static VRPs [23] vehicle routes are designed at the beginning of the planning horizon, *before* uncertain data become known. Uncertainty may affect which service requests are present, user demands, user service times or travel times. If input data are uncertain, it is usually impossible to satisfy the constraints for all realisations of the random variables. If uncertainty affects the constraints but the objective function is deterministic, it can be required that constraints be satisfied with a given probability (chance constrained programming, CCP). In a more general approach, a first phase solution is constructed *before* uncertain data are available and corrective (or *recourse*) actions are taken at a second stage once all the realisations of the random variables become known. The objective to be minimized is the first stage cost plus the expected recourse cost (stochastic programming with recourse, SPR).

(ii) *Dynamic VRPs:* A dynamic problem can also be deterministic or stochastic [24]. In deterministic and dynamic problems, all data are known in advance and some elements of information depend on time. For instance, the VRP with time windows reviewed in [25] belongs to this class of problems. Similarly, the traveling salesman problem (TSP) with time-dependent travel times [26] is deterministic and dynamic. In this problem, a traveling salesperson has to find the shortest closed tour among several cities passing through all cities exactly once, and travel times may vary throughout the day. Finally, in stochastic and dynamic problems (also known as *real-time* routing and dispatching problems) uncertain data are represented by *stochastic processes*. For instance, user requests can behave as a Poisson process (as in [27]). Since uncertain data are gradually revealed during the operational interval, routes are *not* constructed beforehand. Instead, user requests are dispatched to vehicles in an on-going fashion as new data arrive [28].

The *events* that lead to a plan modification can be: (i) the arrival of new user requests, (ii) the arrival of a vehicle at a destination, (iii) the update of travel times. Every event must be processed according to the policies set by the vehicle fleet operator. As a rule, when a new request is received, one must decide whether it can be serviced on the same day, or whether it must be delayed or rejected. If the request is accepted, it is temporarily assigned to a position in a vehicle route. The request is effectively serviced as planned if no other event occurs in the meantime. Otherwise, it can be assigned to a different position in the same vehicle route, or even dispatched to a different vehicle. It is worth noting that at any time each driver just needs to know his next stop. Hence, when a vehicle reaches a destination it has to be assigned a new destination. Because of the difficulty of estimating the current position of a moving vehicle, reassignments could not easily made until quite recently. However, due to advances in communication technologies, route diversions and reassignments are now a feasible option and should take place if this results in a cost saving or in an improved service level [29,30]. Finally, if an improved estimation of vehicle travel times is available, it may be useful to modify the current routes or even the decision of accepting a request or not. For example, if an unexpected traffic jam occurs, some user services can be deferred. It is worth noting that when the demand rate is low, it is useful to relocate idle vehicles in order to anticipate future demands or to escape a forecasted traffic congestion.

2.2. Particular features

As pointed out by Psaraftis [28,31] dynamic VRPs possess a number of peculiar features, some of which have just been described. In the following, the remaining characteristics are outlined.

(i) *Quick response:* Real-time routing and dispatching algorithms must provide a quick response so that route modifications can be transmitted timely to the fleet. To this end, two approaches can be used: simple policies (like the first-come first served (FCFS) policy [27]), or more involved algorithms running on parallel hardware (like the tabu search (TS) heuristics described in [11,12]). As will be explained, the choice between them depends mainly on the objective, the degree of dynamism and the demand rate.

(ii) *Denied or deferred service:* In some applications it is valid to deny service to some users, or to forward them to a competitor, in order to avoid excessive delays or unacceptable costs. For instance, in [11] requests that cannot be serviced with a given time windows are rejected. When no time windows are imposed, some user requests can be postponed indefinitely because of their unfavorable location. This phenomenon can be avoided by imposing dummy time windows, or by adding a nonlinear delay penalty to the objective function.

(iii) *Congestion:* If the demand rate exceeds a given threshold, the system becomes saturated, i.e., the expected waiting time of a request grows to infinity.

2.3. The degree of dynamism of a problem

Designing a real-time routing algorithm depends to a large extent on how much the problem is dynamic. To quantify this concept, Lund et al. [32] and Larsen [10] have defined the *degree of dynamism* of a problem.

Without loss of generality, we assume that the planning horizon is a given interval $[0, T]$, possibly divided into a finite number of smaller intervals. Let n_s and n_d be the number of static and dynamic requests, respectively. Moreover, let $t_i \in [0, T]$ be the *occurrence time* of service request i. Static requests are such that $t_i = 0$ while dynamic ones have $t_i \in (0, T]$. Lund et al. [32] define the degree of dynamism δ as

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

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

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

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

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

It can be shown that δ'' also varies between 0 and 1. Moreover, if no time windows are imposed (i.e., $a_i = t_i$ and $b_i = T$), then $\delta'' = \delta'$. As a rule, vendor-based distribution systems (such as those distributing heating oil) are weakly dynamic. Problems faced by long-distance couriers and appliance repair service companies are moderately dynamic [33] . Finally, emergency services and taxi cab services exhibit a strong dynamic behavior.

2.4. Objectives

In real-time routing problems the objective to be optimized is often a combination of different measures. Larsen [10] observes that in weakly dynamic systems the focus is on minimizing routing cost. On

the other hand, when operating a strongly dynamic system, minimizing the expected *response time* (i.e., the expected time lag between the instant a user service begins and its occurrence time) becomes a key issue. Another meaningful criterion which is often considered (alone or combined with other measures) is throughput optimization, i.e., the maximization of the expected number of requests serviced within a given period of time [28]. A completely different approach is followed in [34,35]. In particular, the former paper is based on the assumption that in most real-time routing problems the objective is fuzzy. Hence, a suitably trained neural network is used to reproduce automatically the dispatching decisions of skilled personnel.

2.5. Spatial distribution and time pattern of user requests

As explained later in this section, better real-time dispatching and routing decisions can be made if uncertain data estimations (derived from historical data) are used. In several papers (see, e.g., [27,36–38]), it is supposed that user requests are uniformly distributed in a convex bounded Euclidean region, and occur according to a Poisson process with arrival rate λ. In problems with pick-ups and deliveries (see, e.g., [37]) it is also assumed that the delivery locations are independent of the pick-up locations.

2.6. Comparing different dispatching and routing procedures

Evaluating a dispatching and routing algorithm can be done analytically if specific assumptions are satisfied (see, e.g., [27] where demands are modeled as a Poisson process). If these hypotheses do not hold, algorithmic performances have to be evaluated empirically through discrete-time simulation, as in [11]. Another way to assess the performance of a heuristic for a dynamic routing problem is to run it independently on the corresponding static data assuming all information is available when planning takes place. By comparing the static and dynamic solution one can compute the *value of information* as is typically done in decision trees. Such an approach was recently used by Mitrović-Minić et al. [39].

3. Sequential algorithms

Sequential algorithms can be broadly divided into three main categories: simple policies, classical insertion procedures and metaheuristics.

3.1. Simple policies

Routing policies are simple rules applied repeatedly in order to dispatch requests to vehicles and build routes. In [27,38] several policies have been proposed for the dynamic traveling repairman problem (DTRP), a stochastic and dynamic variant of the classical TSP. Similarly, in [36] some policies for the stochastic and dynamic version of the capacitated VRP have been illustrated and analyzed. Finally, in [37] the results in [27] are generalized to a real-time single vehicle problem with pick and delivery. We now outline some classical policies for the DTRP. The DTRP objective is then to minimize the expected time the demand spends in the system, as opposed to the expected distance that the vehicle travels.

(i) *First come first served policy:* Demands are served in the order in which they are received by the dispatcher.
(ii) *Stochastic queue median (SQM) policy:* This is a modification of the FCFS policy in which the vehicle is based at the stochastic median of the service region. When a call arrives, the vehicle travels from the median to the user. When the service is finished, the vehicle returns to the median.
(iii) *Nearest neighbor (NN) policy:* When the current user service is over, the vehicle serves the nearest unserviced request.
(iv) *TSP policy:* Requests are batched into sets of a given size. Once a set of requests has been collected, a TSP is solved. The requests are then serviced according to the optimal TSP solution.

These policies can be studied analytically. If the requests occur according to a Poisson process with arrival rate λ and user locations are uniformly distributed in an Euclidean region, then the SQM policy yields an asymptotically optimal

performance in light traffic ($\lambda \to 0$). Unfortunately, the SQM policy becomes very soon unstable as demand rate increases. For this reason, Papastavrou [38] has devised a new routing policy, called the *generation policy* that performs optimally in light traffic, and reasonably well in heavy traffic.

3.2. Classical insertion procedures

Instead of using simple policies that perform well under specific hypotheses, more involved heuristics having good empirical performance in several operational conditions can be employed. At present such procedures simply re-optimize the vehicle routes when new data become available, without any look-ahead. *Insertion procedures* in which real-time customers are inserted in the best position of the current routes have been described in [18]. Such heuristics are simple and can be run very fast even on sequential hardware.

Insertion algorithms can be executed with a *rolling horizon*, as proposed by Psaraftis [28] for a sealift emergency problem in which cargos are moved by ship. At time t only the requests with pick-up time included in $[t, t+\tau]$ are considered, τ being the duration of the rolling horizon. Moreover, cargoes with pick-up time in $[t, t+\alpha\tau]$, $0 \leqslant \alpha \leqslant 1$, are dispatched definitively to a ship.

Recently Mitrović-Minić et al. [39] have extended the concept of rolling horizon to that of double horizon. Whenever a new request arises, special effort is placed on optimizing the short term part of the current solution (corresponding in general to the period containing the pick-up of a parcel) while less effort is put on the distant future (corresponding in general to the period containing the delivery of the parcel). The idea behind this approach is that the later portions of the vehicle routes are likely to be subjected to several changes as time evolves and there is no point in optimizing them too finely at an early stage.

3.3. Metaheuristics

Unfortunately simple insertion procedures not always provide good solutions. In order to generate better vehicle routes, it is customary to use metaheuristics. These are iterative procedures aimed at finding near-optimal solutions for large-scale combinatorial optimization problems [40]. They include simulated and deterministic annealing, genetic algorithms, neural networks [34,41] and expert systems [42], ant colony methods, TS, adaptive memory (AM) techniques and variable neighborhood search (VNS). Among them, the last three algorithms (especially when jointly used) have proven the most successful methods. TS is a local search procedure that iteratively moves from a current solution to its best neighbor even if this causes a deterioration in the objective function value [43]. To avoid cycling a short term memory (like a tabu list or tabu attributes) is used. In addition, a diversification and an intensification phase can be used to visit new regions of the solution space, and to explore the more promising regions, respectively. AM based procedures implement some ideas taken from genetic algorithms [44]. High quality routes provided by several runs of constructive heuristics or by metaheuristics are stored, selected and combined in order to generate new solutions. VNS uses a systematic change of neighborhood during local search [45]. TS, AM and VNS can be combined in several ways in order to generate better solutions, although this usually comes at expense of a larger computation time. For instance, TS can be seen as an improved local search tool in a VNS scheme and, vice versa, VNS can be used to implement intensification and diversification in TS.

4. Parallel algorithms

When using metaheuristics to solve a real-time routing and dispatching problem, a parallel implementation is usually needed in order to make route re-optimization computation time acceptable. The parallelization of metaheuristics can be accomplished in a number of ways depending on problem structure and hardware at hand. In what follows we examine TS based procedures.

4.1. Tabu search based parallelization strategies

Crainic et al. [46] classify parallel TS procedures according to three criteria: *search control cardinality*, *search control type* and *search differentiation*.

The first criterion indicates whether the control of the parallel search is performed by a single processor (1-control, 1-C), or distributed among several processors (p-control, p-C). The second measure denotes the way (*synchronous* or *asynchronous*) communication is performed among processors. Synchronous communication can be independent of computation status (rigid synchronization, RS) or not (knowledge-based synchronization, KS). In asynchronous communication, a processor that finds a new best solution broadcasts a message to the other processors. In the simplest case, the single solution is sent (collegial communication, C) while in knowledge-based collegial communication (KC) additional information are transmitted to the receivers. Finally, the third criterion accounts for the way different searches are carried on. Four alternatives are available: single initial point–single strategy (SPSS) if a single search is performed; single initial point–multiple strategies (SPMS) when each processor carries out a different search starting from the same initial solution; multiple initial points–single strategy (MPSS) if each processor performs the same search starting from different initial solutions; multiple initial points–multiple strategies (MPMS) if each processor is in charge of a different search starting from a different initial solution. In SPMS and MPMS, the searches may be performed by different algorithms or, more commonly, by the same algorithm with different parameters. Not every combination of such parameters is feasible (e.g., 1-C/C/MPMS does not make any sense). The most common configurations are

(i) *1C/RS/SPSS* (*or master–slave*) *strategy:* A single search is performed by a single processor (called *master*) which dispatches time-consuming tasks to the other processors (called *slaves*). There is no communication among the slaves.
(ii) *1C/KS/SPSS strategy:* Compared to 1C/RS/SPSS, the slaves (which still do not communicate among themselves) can be required by the master to stop computing. This is usually the case when each slave is assigned a relatively simple search.
(iii) *p-C/RS/MPSS*, *p-C/RS/SPMS and p-C/RS/MPMS strategies:* Several independent searches (with different initial points or parameters) are executed in parallel. There is no communication among processors except at the end of the computations when the best solution is selected.
(iv) *p-C/C/SPMS*, *p-C/C/MPSS and p-C/C/MPMS strategies:* Each processor performs a different search. Once a processor finds a new best solution, it sends it to the other processors that re-initialise their searches.

4.2. Parallelization strategies for DVRPs

When choosing a parallelization strategy for a dynamic and stochastic VRP, three main factors should be taken into account: (i) the computational effort required to generate a new feasible near-optimal solution when a request occurs, (ii) the overall computation power available, (iii) the average inter-arrival time of service requests. The first two parameters determine how fast route re-optimization can be done.

Another important issue influencing the design and efficiency of parallel algorithms lies in the architecture of the target parallel machine on which the implementation will be run. In what follows, we assume the availability of a coarse-grained parallel environment (i.e., a networks of computers). This choice is motivated by the fact that one of the leading tendencies among the current trends in parallel computing is to move away from specialized traditional supercomputing platforms to cheaper and general purpose systems consisting of shared memory multiprocessor machines or single PCs or workstations connected through fast local networks or very high speed networks [47]. The recent growth in the use of clusters of computers is mainly related to the good cost/performance ratios of such systems when compared to other parallel machines. Furthermore, these ''distributed supercomputers'' are very flexible since additional computing and communication capacity can be easily configurated into the system.

In order to describe the parallelization strategies, we refer to a *master–slave* computational model, by far the most common for distributed computing. In the case where the problem under investigation is characterized by large

neighborhoods and the evaluation and execution of each move requires a relatively small computing effort, the 1C/RS/SPSS parallelization strategy represents an efficient way of generating a new feasible near-optimal solution when a new request occurs. In this strategy, the neighborhood of the current solution is partitioned into as many sets as the number of available processors and the evaluation of each set is executed concurrently by the slaves. At the end of each iteration, the master processes the information resulting from the slave operations, performs either the best move (i.e., *single-step* parallelization) or a sequence of consecutive moves (i.e., *multiple-step* parallelization) and restarts the computation by assigning to each salve a new set of neighbor solutions to be evaluated in parallel. A *look-ahead* implementation is also possible; in this case the slaves perform a predetermined number of iterations before the synchronization phase is executed. A look-ahead implementation is preferable since it is characterized by fewer synchronization events and lower interprocess communication and is thus more suitable to efficient implementation on a cluster of computers.

In the case of dynamic and stochastic VRPs, for which numerous iterations may be performed in a relatively limited amount of time (e.g., thousands of iterations in a few seconds), *domain decomposition* parallelization strategies represent valid approaches to design efficient parallel algorithms for route re-optimization. More specifically, the feasible domain of the problem is partitioned into several disjoint sets and different processors acting in parallel execute a predetermined number of iterations of the TS heuristic on each subset. After synchronization, the solutions provided by the slaves are merged appropriately to obtain a new feasible solution. The strategy just outlined can be used efficiently also in the case in which the problem under investigation is characterized by large neighborhoods and the search for the best move at each iteration is a computationally intensive task.

Multithread parallelization strategies represent alternative approaches for the parallel re-optimization of the routes when a new request is received. These are characterized by the execution of several search processes (i.e., threads) that investigate the same solution space in parallel, starting from possibly different initial solutions and using TS strategies with possibly different parameter settings. These threads can be either *independent* or *cooperative*.

In the first case, the threads do not exchange any information and there is no attempt to take advantage of the multiple trajectories determined during the search of the solution space. Indeed, after each thread has terminated its search, they synchronize and the master selects the best overall solution which can be used as starting point for a new round of independent searches. Thus, independent-threads parallel methods are equivalent to multistart sequential heuristics.

In a cooperative-threads approach, the threads cooperate, exchange and share information and the information collected along each trajectory is used to improve the other trajectories. Thus, it is possible not only to speed-up the convergence to the best solution, but also to find better solutions than independent-threads strategies within the same computation time.

It is worth observing that different strategies can be used to implement cooperation between threads. In the case of a network of computers, the shared information can be implemented as a pool of solutions in the local memory of the master process responsible for pool management. All communications take place between the master and individual threads, whereas there are no direct communications among the search processes. In addition, the threads are in charge of controlling the access to shared knowledge.

On the basis of the information stored in the memory and its use, it is possible to define adaptive [44] and central memory strategies [46]. In the parallelization strategies described above, one of the most difficult aspect to be set up is the determination of the number of iterations for which the algorithm is run. This number should be set by considering the temporal distribution of the immediate requests for service.

5. Conclusions

Although tens of papers have been devoted to real-time VRPs, the literature is still disorganized

and a few major issues have not been addressed yet. In our opinion, the following future research directions can be drawn.

5.1. General issues

(i) Heuristics with some look-ahead capability should be developed. To highlight this aspect, suppose, for instance, there are very few requests at the begin of the planning horizon. Should we serve them by using the *least number* of vehicles? If so, should the idle vehicles be relocated? And where? Or should we serve such requests using the *maximum number* of vehicles (so that their service is completed as soon as possible)? The best choice mainly depends on future demand and on future travel times. Hence, it may be wiser to incorporate a look-ahead feature into route building.

(ii) New metaheuristics, capable of providing good solutions as the degree of dynamism δ'' changes should be developed. For $\delta'' = 0$ such procedures should work as static VRP algorithms, while for $\delta'' \to 1$ they should show proper behavior both in light and heavy traffic.

5.2. Specific issues

(i) Routing and dispatching problems where travel times vary in real-time will gain in importance with the development of intelligent transportation systems (ITS). However, to our knowledge, no research has been devoted to this topic. The article by Malandraki and Daskin [26] where efficient heuristics are presented for the static case should be extended to the real-time case.

(ii) Booking problems (defined over a multiple day horizon) like [14] should studied in greater detail.

(iii) Some dynamic ambulance allocation and relocation problems deserve more attention (see [16]).

(iv) Automated guided vehicle (AGV) dispatching and routing problems arising in manufacturing plants and port terminals should be addressed. A specific aspect of these problems is *track contention* [48].

(v) Real-time arc routing problems whose applications are related to road gritting and snow collection are other interesting research topics.

Acknowledgements

This research was partially supported by the Italian National Research Council, by the Center of Excellence for High-Performance Computing, University of Calabria, Italy, and by the Canadian Natural Sciences and Engineering Research Council under grant OGP0039682. This support is gratefully acknowledged.

References

[1] P. Toth, D. Vigo (Eds.), The Vehicle Routing Problem, in: SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 2002.

[2] G.G. Brown, G.W. Graves, Real-time dispatch of petroleum tank trucks, Management Science 27 (1981) 19–32.

[3] G.G. Brown, C. Ellis, G.W. Graves, D. Ronen, Real-time, wide area dispatching of Mobil tank trucks, Interfaces 17 (1) (1987) 107–120.

[4] W.B. Powell, A stochastic model of the dynamic vehicle allocation problem, Transportation Science 20 (1986) 117–129.

[5] M. Goetschalckx, A decision support system for dynamic truck dipatching, International Journal of Physical Distribution and Materials Management 14 (1988) 34–42.

[6] W.B. Powell, Real-time optimization for truckload motor carriers, OR/MS Today 18 (1990) 28–33.

[7] C. Rego, C. Roucairol, Using tabu search for solving a dynamic multi-terminal truck dispatching problem, European Journal of Operational Research 83 (1995) 411–429.

[8] M.W.P. Savelsbergh, M. Sol, Drive: Dynamic routing of independent vehicles, Operations Research 46 (1998) 474–490.

[9] A. Campbell, L. Clarke, A. Kleywegt, M. Savelsbergh, The inventory routing problem, in: G. Laporte, T.G. Crainic (Eds.), Fleet Management and Logistics, Kluwer, Boston, 1998.

[10] A. Larsen, The dynamic vehicle routing problem, Ph.D. Thesis, Institute of Mathematical Modelling, Technical University of Denmark, 2001.

[11] M. Gendreau, F. Guertin, J.-Y. Potvin, É. Taillard, Parallel tabu search for real-time vehicle routing and dispatching, Transportation Science 33 (1999) 381–390.

[12] M. Gendreau, F. Guertin, J.-Y. Potvin, R. Séguin, Neighbourhood search heuristics for a dynamic vehicle

dispatching problem with pick-ups and deliveries, Technical report CRT-98-10, Centre de Recherche sur les Transports, Université de Montréal, 1998.
[13] J.-F. Cordeau, G. Ghiani, G. Laporte, R. Musmanno, A parallel tabu search algorithm for the static dial-a-ride problem, working paper, 2002.
[14] O.B.G. Madsen, K. Tosti, J. Vælds, A heuristic method for dispatching repair men, Annals of Operations Research 61 (1995) 213–226.
[15] A. Weintraub, J. Aboud, C. Fernandez, G. Laporte, E. Ramirez, An emergency vehicle dispatching system for an electric utility in Chile, Journal of the Operational Research Society 50 (1999) 690–696.
[16] L. Brotcorne, G. Laporte, F. Semet, Ambulance location and relocation models, European Journal of Operational Research 147 (3) (2003) 451–463.
[17] J.F. Cordeau, G. Laporte, A tabu search heuristic for the static multi-vehicle dial-a-ride problem, Technical Report, May, 2002.
[18] S. Roy, J.-M. Rousseau, G. Lapalme, J.A. Ferland, Routing and scheduling of transportation services for disabled: Summary report, Technical Report, Centre de Recherche sur les Transports, Université de Montréal, 1984.
[19] O.B.G. Madsen, H.F. Ravn, J.M. Rygaard, A heuristic algorithm for a dial-ride problem with time windows, Annals of Operations Research 60 (1995) 193–208.
[20] H.N. Psaraftis, A dynamic programming solution to the single many-to-many immediate request dial-a-ride problem, Transportation Science 14 (1980) 130–154.
[21] M. Gendreau, G. Laporte, F. Semet, Solving an ambulance location model by tabu search, Location Science 2 (1997) 75–88.
[22] M. Gendreau, G. Laporte, F. Semet, A dynamic model and parallel tabu search heuristic for real-time ambulance relocation, Parallel Computing 27 (2001) 1641–1653.
[23] G. Laporte, F.V. Louveaux, Solving stochastic routing problems, in: T.G. Crainic, G. Laporte (Eds.), Fleet Management and Logistics, Kluwer, Boston, 1998, pp. 159–167.
[24] W.B. Powell, P. Jaillet, A.R. Odoni, Stochastic and dynamic networks and routing, in: M.O. Ball, T.L. Magnanti, C.L. Monma, G.L. Nemhauser (Eds.), Handbooks in Operations Research and Management Science, 8: Network Routing, Elsevier Science, Amsterdam, 1995, pp. 141–295.
[25] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M.M. Solomon, F. Soumis, The vehicle routing problem with time windows, in: P. Toth, D. Vigo (Eds.), The Vehicle Routing Problem, in: SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 2001, pp. 157–193.
[26] C. Malandraki, M.S. Daskin, Time dependent vehicle routing problems: Formulations, properties, and heuristics algorithms, Transportation Science 26 (1992) 185–200.
[27] D. Bertsimas, G.J. van Ryzin, A stochastic and dynamic vehicle routing problem in the Euclidean plane, Operations Research 39 (1991) 601–615.
[28] H.N. Psaraftis, Dynamic vehicle routing problems, in: B.L. Golden, A.A. Assad (Eds.), Vehicle Routing: Methods and Studies, Elsevier Science, Amsterdam, 1998, pp. 223–248.
[29] M. Gendreau, J.-Y. Potvin, Dynamic vehicle routing and dispatching, in: T.G. Crainic, G. Laporte (Eds.), Fleet Management and Logistics, Kluwer, Boston, 1998, pp. 115–126.
[30] S. Ichoua, M. Gendreau, J.-Y. Potvin, Diversion issues in real-time vehicle dispatching, Transportation Science 34 (2000) 426–435.
[31] H.N. Psaraftis, Dynamic vehicle routing: Status and prospects, Annals of Operations Research 61 (1995) 143–164.
[32] K. Lund, O.B.G. Madsen, J.M. Rygaard, Vehicle routing problems with varying degrees of dynamism, Technical report, Institute of Mathematical Modelling, Technical University of Denmark, 1996.
[33] A. Larsen, O.B.G. Madsen, M.M. Solomon, Partially dynamic vehicle routing—models and algorithms, Technical report, Institute of Mathematical Modelling, Technical University of Denmark, 1999.
[34] J.-Y. Potvin, Y. Shen, J.-M. Rousseau, Neural networks for automated vehicle dispatching, Computers & Operations Research 19 (1992) 267–276.
[35] J.-Y. Potvin, G. Dufour, J.-M. Rousseau, Learning vehicle dispatching with linear programming models, Computers & Operations Research 20 (1993) 371–380.
[36] D. Bertsimas, G.J. van Ryzin, Stochastic and dynamic vehicle routing problem in the Euclidean plane with multiple capacitated vehicles, Operations Research 41 (1993) 60–76.
[37] M.R. Swihart, J.D. Papastavrou, A stochastic and dynamic model for the single-vehicle pick-up and delivery problem, European Journal of Operational Research 114 (1999) 447–464.
[38] J. Papastavrou, A stochatic and dynamic routing policy using branching processes with state dependent immigration, European Journal of Operational Research 95 (1996) 167–177.
[39] S. Mitrović-Minić, R. Krishnamurti, G. Laporte, The double-horizon heuristic for the dynamic pickup and delivery problem with time windows, submitted for publication.
[40] I.H. Osman, J.P. Kelly (Eds.), Meta-heuristics: Theory and Applications, Kluwer, Boston, 1996.
[41] Y. Shen, J.-Y. Potvin, J.-M. Rousseau, S. Roy, A computer assistant for vehicle dispatching with learning capacities, Annals of Operations Research 61 (1995) 189–211.
[42] P.K. Bagchi, B.N. Nag, Dynamic vehicle scheduling: An expert systems approach, International Journal of Physical Distribution and Logistics Management 21 (1991) 10–18.
[43] F. Glover, Future paths for integer programming and links to artificial intelligence, Computers & Operations Research 5 (1986) 533–549.
[44] Y. Rochat, É. Taillard, Probabilistic diversification and intensification in local search for vehicle routing, Journal of Heuristics 1 (1995) 147–167.

[45] N. Mladenović, P. Hansen, Variable neighbourhood search, Computers & Operations Research 24 (1997) 1097–1100.
[46] T.G. Crainic, M. Toulouse, M. Gendreau, Towards a taxonomy of parallel tabu search algorithm, INFORMS Journal on Computing 9 (1997) 61–72.
[47] R. Buyya (Ed.), High Performance Cluster Computing: Architectures and Systems, vols. 1 and 2, Prentice-Hall, Englewood Cliffs, NJ, 1999.
[48] P. Caricato, G. Ghiani, A. Grieco, E. Guerriero, Parallel tabu search for a vehicle rounting problem under track contention, Parallel Computing, forthcoming.