European Journal of Operational Research 202 (2010) 8–15

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Invited Review

Dynamic pickup and delivery problems

Gerardo Berbeglia [a], Jean-François Cordeau [b], Gilbert Laporte [a,*]

[a] Canada Research Chair in Distribution Management, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Montréal, Canada H3T 2A7
[b] Canada Research Chair in Logistics and Transportation, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Montréal, Canada H3T 2A7

ARTICLE INFO

Article history:
Received 31 October 2008
Accepted 27 April 2009
Available online 6 May 2009

Keywords:
Vehicle routing
Dynamic
Stacker crane
Dial-a-ride problem
Pickup and delivery

ABSTRACT

In the last decade, there has been an increasing body of research in dynamic vehicle routing problems. This article surveys the subclass of those problems called dynamic pickup and delivery problems, in which objects or people have to be collected and delivered in real-time. It discusses some general issues as well as solution strategies.

1. Introduction

Pickup and delivery problems (PDPs) are a class of vehicle routing problems in which objects or people have to be transported between an origin and a destination. They can be classified into three different groups. The first group consists of *many-to-many* problems, in which any vertex can serve as a source or as a destination for any commodity. An example of a many-to-many problem is the *Swapping Problem* (Anily and Hassin, 1992). In this problem, every vertex may initially contain an object of a known type of commodity as well as a desired type of commodity. The problem consists of constructing a route performing the pickups and deliveries of the objects in such a way that at the end of the route, every vertex possesses an object of the desired type of commodity. Problems in the second group are called *one-to-many-to-one* problems. In these problems commodities are initially available at the depot and are destined to the customer vertices; in addition, commodities available at the customers are destined to the depot. Finally, in *one-to-one* problems, each commodity (which can be seen as a request) has a given origin and a given destination. Problems of this type arise, for example, in courier operations and door-to-door transportation services.

A routing problem is said to be *static* when all the input data of the problem are known before routes are constructed. In a *dynamic* routing problem, some of the input data are revealed or updated during the period of time in which operations take place. In a dynamic pickup and delivery problem, the input data which are revealed over time are generally the user requests. In contrast to what happens in a static problem, the planning horizon of a dynamic problem may be unbounded. Therefore, a solution to a dynamic problem cannot be a static output, but rather a solution strategy which, using the revealed information, specifies which actions must be performed as time goes by.

Most studies of pickup and delivery problems have focused on the static case and little work was done on the dynamic counterpart of the PDPs. One example of a dynamic PDP appears in the transportation of handicapped and elderly people in urban areas. The dynamic aspect in this problem, which is called the *dial-a-ride problem* (DARP), comes from the fact that transportation requests are sometimes received the same day they need to be served. Another example arises in the same day transportation services of letters or parcels performed by courier companies. In this problem, most requests are received on their service day. These are two examples of dynamic one-to-one PDPs. We are not aware of any studies related to dynamic many-to-many PDPs or dynamic one-to-many-to-one PDPs.

In this paper we provide a general framework for dynamic one-to-one PDPs, we describe solution concepts and, without being exhaustive, we present the most relevant literature on dynamic pickup and delivery problems. The paper is organized as follows. In Section 2, we present the framework itself we explain the actions that any system must apply to solve them. General solution concepts for dynamic vehicle routing problems are introduced in Section 3. Sections 4–6 survey the *dynamic stacker crane problem*, the *dynamic vehicle routing problem with pickups and deliveries*, and the *dynamic dial-a-ride problem*, respectively.

* Corresponding author.
E-mail addresses: gerardo@crt.umontreal.ca (G. Berbeglia), cordeau@crt.umontreal.ca (J.-F. Cordeau), gilbert@crt.umontreal.ca, Gilbert.Laporte@hec.ca (G. Laporte).

0377-2217/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2009.04.024

These problems are the dynamic counterparts of the one-to-one static PDPs surveyed in Berbeglia et al. (2007). Finally, some conclusions are provided in Section 7.

2. A framework for dynamic one-to-one pickup and delivery problems

We now formalize the framework used for the study of dynamic pickup and delivery problems. A *request* consists of a demand of a certain load for a transportation service from an origin vertex to a destination vertex. The origin (destination) vertex of request i is denoted by i^+ (i^-), and the load is denoted by d_i. Let R denote the set of potential requests. Pickup and delivery problems are defined on a complete and directed graph $G = (V,A)$ with vertex set $V = \{0\} \cup \{i^+ \mid i \in R\} \cup \{i^- \mid i \in R\}$, i.e., V contains the origin and destination of all the requests, in addition to vertex 0 which represents the depot. The arc set is defined as $A = \{(i,j) : i,j \in V, i \neq j\}$. Each arc $(i,j) \in A$ has a non-negative length or cost c_{ij} and a non-negative travel time t_{ij}. A *route* is a circuit over some vertices, starting and finishing at the depot. The pickup and delivery vertices of a request cannot be visited by different routes and the pickup vertex naturally precedes the delivery vertex.

At each time instant t, each vehicle $k \in K$ is either serving a vertex, waiting at a vertex, or moving toward a vertex. In either of these cases, the request associated with the vertex is known at time t.

The decisions that need to be taken can be divided into two classes of *decision epochs*: (i) *wait or go* and (ii) *accept or reject*. A decision epoch of type *wait or go* occurs when a vehicle finishes serving a vertex. Assuming the actual time is t, the system has to choose between waiting at the vehicle's current location or move toward a vertex whose request is known at time t. If the decision *wait* has been chosen, a series of decision epochs of type *wait or go* will be triggered one at each time step until the decision *go* has been chosen specifying the next vertex to visit. Finally, throughout the time horizon, service requests are received. Such events trigger *accept or reject* decision epochs at which the system has to decide whether or not to accept the request.

A *dynamic one-to-one pickup and delivery algorithm* consists of a system that decides which actions to perform at decision epochs of type *wait or go* and *accept or reject*. Typically, all the requests known in advance and all accepted requests must be served by the given fleet of vehicles. The quality of such an algorithm can be measured by the number or proportion of accepted requests, the total distance traveled by the vehicles, and the time taken to decide whether or not to accept a request. Additional assumptions and constraints on vehicle routes, such as time windows and maximum ride times may be present depending on the application considered.

The literature on *dynamic one-to-one pickup and delivery problems* can be partitioned into three different classes of problems. In the case where requests are for the transportation of objects and the vehicles can serve more than one request at a time, the problem is typically called the *dynamic vehicle routing problem with pickups and deliveries* (Dynamic VRPPD). When the vehicles can only perform one request at a time (e.g., when all requests have a load equal to the vehicle capacity), the problem is called the *dynamic stacker crane problem* (Dynamic SCP). Finally, in the case where the transportation requests consists of passengers the problem is called the *dynamic dial-a-ride problem* (Dynamic DARP). It could be argued that this problem is the same as the Dynamic VRPPD. However, the Dynamic DARP differs from the Dynamic VRPPD because of its tight time windows and maximum ride time constraints which are generally imposed to ensure a good quality of service to users.

3. General solution concepts

While this paper focuses on dynamic pickup and delivery problems, some of the solution concepts and features also apply to dynamic vehicle routing problems in general. In this section we discuss the following features: basic solution strategies, algorithmic performance assessment, the use of information about future requests, vehicle diversion, and the degree of dynamism. Some of these and others are also discussed in Psaraftis (1988) and Ghiani et al. (2003).

3.1. Basic solution strategies

As already stated, in *dynamic* problems, the information is gradually revealed over time. A basic and commonly used strategy for solving a dynamic vehicle routing problem is to adapt an algorithm that solves the static version of the problem. Two approaches can be distinguished.

The first one consists of solving a static problem each time new information (such as a new request or a cancellation) is revealed. Under such a strategy, the static algorithm may need to be modified to ensure that its solution is feasible with respect to past decisions. One important drawback of this strategy is that performing a complete reoptimization every time new information is revealed may be too time consuming, and therefore inadequate for a real-time setting. The second approach, which is the one generally used, is the following. The static algorithm is applied only once at the beginning of the planning horizon to obtain an initial solution with the available information. When new information is revealed, the current solution is *updated* with heuristic methods such as insertion heuristics, deletion heuristics, and interchange moves, sometimes coupled with a local search algorithm. These update mechanisms are generally sufficiently fast to be used in real-time. In the intervals elapsed between the time instants at which new information is revealed, some more robust optimization methods are sometimes applied to improve the current solution.

3.2. Algorithmic performance assessment

There are several ways of assessing the performance of algorithms for an online (or dynamic) combinatorial optimization problem. One of these is *competitive analysis* (Borodin and El-Yaniv, 2005). Consider a minimization problem P and let I denote the input set. We denote by $Opt(i)$ the cost value of an optimal solution for the instance $i \in I$. In the *online* version of the problem P, the input is received in a real-time fashion and the output must be produced online as well. Consider any online algorithm A for solving the online version of problem P, and let $A(i)$ be the cost value of the solution obtained by A when given the instance $i \in I$. We say that the online algorithm A is *c-competitive* if there is a constant α such that

$$A(i) \leqslant c \cdot Opt(i) + \alpha \quad \forall i \in I. \tag{1}$$

Whenever α is equal to zero, the algorithm A is said to be *strictly-c-competitive*. Thus, competitive analysis offers a worst-case measure of performance. For example, an online algorithm which is *strictly-2-competitive* guarantees that for every instance of the problem, the value of the solution returned would never be more than twice the value of the optimal solution of the static problem. Since it is generally hard to perform this type of analysis for complex problems and algorithms, the literature on competitive analysis for dynamic pickup and delivery problems focuses on very restricted cases. For instance, in these studies demands are unitary and time windows are generally not present. It is worth observing that stochastic information about the future is not

considered in competitive analysis. We review some of the most important studies on competitive analysis for pickup and delivery problems in Section 5.1.

Many articles presenting an algorithm for a dynamic PDP assess the algorithm's performance using a less strict analysis. For example, in Mitrović-Minić et al. (2004), the authors have analyzed the performance of a dynamic algorithm over a finite subset of the input instances I. That is, they have shown how far away solutions returned by the dynamic algorithm are with respect to their associated static problems on a fixed (and finite) set of instances. The authors have used a measure called the *value of information* for evaluating their dynamic algorithms. Given an instance i, an algorithm A for the dynamic problem and an algorithm A_s which is a static version of the algorithm A for the static problem, the *value of information under A* is defined as $(A(i) - A_s(i))/A(i)$. Another way used to evaluate different algorithms for dynamic PDPs is by comparing their relative performance.

3.3. Anticipation of future requests

In many dynamic PDPs and in dynamic vehicle routing problems in general, some information about future requests is known as a probability distribution. For example, the incoming requests may follow a specific Poisson distribution (see, e.g., Swihart and Papastavrou, 1999). Because of the problem complexity, the exact probability distribution of future events is not always known but can be approximated through the use of historical data (Van Hentenryck and Bent, 2004; Hvattum et al., 2006, 2007). These dynamic PDPs in which some information about future request is known are called *dynamic and stochastic PDPs*. Although studies on dynamic and stochastic PDPs are very scarce, there has been in the last few years an increasing effort to determine how stochastic information can be exploited to improve the performance of dynamic strategies for dynamic vehicle routing problems. Van Hentenryck and Bent (2004) present a general framework for solving *online (dynamic) stochastic combinatorial optimization problems* based on scenario sampling with applications to vehicle routing. This method requires two main black-boxes: (i) an optimization algorithm for the static problem, and (ii) a procedure capable of producing a sample of the future events. The authors then propose a family of algorithms that solve online combinatorial optimization problems by iteratively solving a series of static problems obtained by the use of the sampling black-box and integrating the results.

The knowledge that future requests will arrive can also be used as follows. Consider a dynamic PDP whose objective is to minimize the total distance traveled by the vehicles. Assume we have an algorithm for solving this problem which is an adaptation of an algorithm for the static version of the problem. Every time a new request arrives, the current solution is updated either by an efficient insertion method or by complete reoptimization. Will the updating procedure also try to minimize the total distance traveled? If it is known that no more requests will arrive, the answer is yes. However, when it is known that new requests will come, it may be worth modifying this objective. For instance, it may be preferable to emphasize the minimization of the distance traveled in the near future since the route will most probably change in the long term with the arrival of new requests. Mitrović-Minić et al. (2004) have considered the short-term objective of minimizing route length and a long term objective of maximizing the slack time, thus favoring the accommodation of future requests. The modification of the objective function for the updating algorithms is an interesting research topic and it is recurrent in many studies of dynamic PDPs (see, e.g., Savelsbergh and Sol, 1998; Gutenschwager et al., 2004).

Yet another way of taking future requests into account in dynamic PDPs is the use of *waiting strategies*. Given a vehicle route $r = (v_1, \ldots, v_k)$, a *waiting strategy* consists of determining how much time the vehicle will wait at each of the vertices of r before resuming its route. In a dynamic context it may sometimes be beneficial to wait at some vertices in anticipation of future requests and thus reduce the overall traveled distance or maximize the probability of serving a request. Similarly, a *buffering strategy* consists of holding a request for a while before assigning it to a vehicle route. Alternatively, the vehicle could move to another location from which future requests could easily be reached, for example a median computed on the basis of the locations and frequencies of the past requests. Theoretical and experimental results on different problems have shown the importance of waiting strategies (Mitrović-Minić and Laporte, 2004; Branke et al., 2005; Ichoua et al., 2006; Thomas, 2007). Recently, Pureza and Laporte (2008) have shown the advantages of waiting and buffering strategies in an uncapacitated dynamic pickup and delivery problem.

3.4. Vehicle diversion

Most algorithms for dynamic PDPs in particular and for vehicle routing problems in general cannot divert a vehicle from its current destination. The allowance of vehicle diversion makes the driver operations more complex. However, given the arrival of new requests this may be beneficial. The incorporation of diversion in an algorithm brings also some technical difficulties such as how the distances are calculated as well as the problem of estimating the time allocated for the optimization procedure. Ichoua et al. (2000) have studied the latter issue and have devised a diversion algorithm for a dynamic vehicle routing problem. The authors have modified a tabu search algorithm for an uncapacitated dynamic pickup and delivery problem by allowing vehicle diversion whenever a new request arrives. Experimental results have shown that the modified algorithm was able to reduce the number of unserved customers and the total distance traveled when compared to the original heuristic. Diversion strategies were also evaluated in a *dynamic stacker crane problem* studied by Regan et al. (1998).

3.5. Degree of dynamism

In some dynamic PDPs more information about the future is known than in others. For instance, in long distance full truckload services the future is better known than in taxi cab services. Lund et al. (1996) have defined the *degree of dynamism* of a dynamic vehicle routing instance as the ratio between the number of dynamic requests and the total number of requests. Later, Larsen (2001) have defined another measure, called the *effective degree of dynamism* which takes into account how long in advance dynamic requests are known. Consider a planning horizon that starts at time 0 and finishes at time T. Let R be the set of requests, and let t_i and l_i denote the time that request $i \in R$ is known and the latest time request $i \in R$ can be served, respectively. The value t_i is sometimes called the *disclosure date* and it is equal to zero for any static request. The *effective degree of dynamism* (*edod*) is defined as

$$edod = \frac{1}{|R|} \sum_{i \in R} \frac{T - (l_i - t_i)}{T}. \tag{2}$$

Observe that $0 \leqslant edod \leqslant 1$, and the larger the *edod*, the "more dynamic" the instance is. Jaillet and Wagner (2006) have quantified the value of the advanced information given by disclosure dates on a new problem they introduced as the *online TSP with disclosure dates*. Experimental analyses on the value of knowing requests in advance was also performed by Tjokroamidjojo et al. (2006) for a version of the dynamic stacker crane problem. We now survey the literature on some specific PDPs.

4. The dynamic stacker crane problem

In the *dynamic stacker crane problem* (Dynamic SCP), each request has to be transported directly from its pickup location to its delivery location. The restriction that only one request can be handled at a time by a vehicle is generally due to capacity. This problem takes its name from the practical problem of managing crane operations.

The main application of the Dynamic SCP is the problem of operating a trucking fleet to move full truckloads between pickup and delivery locations. For this reason the Dynamic SCP is also called the *dynamic full truckload pickup and delivery problem*. The problem is dynamic because customer requests arrive continuously during operations. The problem involves a planning horizon of several days and generally, around half of the requests received are for same day pickups. For a recent survey of dynamic models for these problems see Powell et al. (2007).

In Powell (1987), a methodology is proposed to model and solve a dynamic SCP encountered in the full truckload carrier industry. In his model, the planning area is divided into regions. Several data are assumed to be available, including the probability distribution of the requests between every pair of regions for each of the days of the planning horizon and the average net revenue per request. The problem is represented as a network flow problem where each node represents a region served on a specific day. The arc set consists of two types of arcs. Type I arcs represent deterministic information about known requests and empty moves while Type II arcs, referred to as *stochastic arcs*, are used to estimate the value of sending an additional vehicle to a specific region at a specific time. In Powell et al. (1988), an extension of this network representation is applied to the maximization of profits of the commercial transport division of a carrier in the United States with more than 5800 trailers. The model, which was run four times a day, was able to increase the company's annual profit by *2.5* million US dollars.

Another real-time multi-vehicle truckload pickup and delivery problem in which no stochastic information is considered is presented by Yang et al. (1998). In this problem, each request has a time window during which a pickup must take place if the request is accepted. The authors have modeled the static problem as an integer linear program. To handle the dynamic component, the static problem is solved many times in a rolling-horizon framework. In their paper, the authors provide a comparison between the strategy of solving the static MIP every time a new request arrives and other less time consuming strategies which limit the size of the solution space. Computational results on instances with four vehicles have shown that while the strategy of solving the static MIP outperformed all the others, some simple strategies produced similar results while requiring much less computational effort.

A similar problem was studied by Yang et al. (2004). In this problem, requests also can be rejected and it is permitted to serve a request beyond its time window by incurring a penalty. The authors have compared three rolling-horizon algorithms considered in Yang et al. (1998), as well as two reoptimization policies. The first consists of optimizing the MIP of the static problem every time a new request arrives. The second policy operates in the same way as the first, except that it uses and assumes some knowledge about the probability distribution of incoming requests, such as the expected distance between two random points obtained from the future pickup and delivery locations. The best results were obtained by the second policy, which proved to be between 1% and 5% superior to the first policy without an increase in the computational time in a series of computational tests on instances with 10 vehicles and 1000 requests.

A study aimed at quantifying the value of knowing the requests in advance was carried out by Tjokroamidjojo et al. (2006). The authors considered a full truckload problem with an horizon of 20 days, soft time windows and the possibility of rejecting requests. In this study, no stochastic information about future requests is considered. The solution strategy consists of solving a static MIP every time a new request becomes known. The different policies are characterized by two parameters (τ, T). The parameter τ represents how many days in advance requests are known, while the parameter T represents how many days in advance assignment decisions are made. Results obtained from a set of 20 instances with 50 requests and 10 or 50 cities have shown that policies $(3,1)$, $(5,1)$, $(5,3)$ performed significantly better than the policies $(1,1)$, $(3,3)$, $(5,5)$. In particular, policy $(5,3)$ performed almost as well as in the case where requests were known at the beginning of the operations.

An agent-based approach for a version of the Dynamic SCP was developed by Mes et al. (2007). These authors have studied a problem, inspired from an automated transportation network using AGVs, with soft time window constraints and in which requests cannot be rejected. An agent-based model attempts to model complex phenomena using a set of individual agents with specific tasks and goals. The authors conceived four types of agents: vehicle agents, fleet agents, job agents, and shipper agents. For every new request, a job agent starts an auction asking all vehicle agents to bid only once. The bid consists of a price, an expected departure time and an expected arrival time. Finally the job agent chooses a bid using the Vickrey mechanism, in which the request is assigned to the highest bidder but at the price of the second highest bidder. Simulation experiments on instances with 20 AGVs and 20 locations showed that the agent-based approach behaved generally better than simple scheduling heuristics.

A complex Dynamic SCP arising from a real-world dispatching problem of an electric monorail system was studied by Gutenschwager et al. (2004). The authors have developed a tabu search and a simulated annealing heuristic capable of responding to strong real-time requirements. The solution strategy consists of solving a static problem several times without using stochastic information about future requests. The objective of the static problem was modified to improve the overall performance of the online problem. The application of the proposed algorithms has reduced the number of electric monorail load carriers from 24 to 22 as well as the average and maximum number of simultaneous open transportation requests.

5. The dynamic vehicle routing problem with pickups and deliveries

In the *dynamic vehicle routing problem with pickups and deliveries* (Dynamic VRPPD) vehicles can serve more than one request at the same time. In this type of problem, requests are placed for the transportation of objects (e.g., letters, loads, parcels, etc.). Generally, when time windows are present, they are not tight. In many of these problems, there are no capacity constraints. An example of a dynamic VRPPD is encountered by express courier companies located in many cities. These companies usually receive hundreds or thousands of dynamic requests in the same day. Requests consist carrying letters or parcels from a pickup point to a delivery point. The first part of this section is dedicated to a review of competitive analysis studies of PDPs. The main problem studied in this area is a simple PDP called the *online dial-a-ride problem* (OlDARP). Despite its close name resemblance with the dynamic DARP, the OlDARP does not consider constraints for minimizing user inconvenience which are present in the DARP such as time windows and maximum ride times. For this reason, we believe it makes sense to review the OlDARP in this section rather than in the section on the Dynamic DARP (Section 6). The second part of this section surveys

heuristic algorithms for more complex problems which better model the real-world.

5.1. Competitive analysis studies

In this section we present some of the most relevant competitive analysis studies on dynamic vehicle routing problems with pickups and deliveries. For a recent survey on competitive analysis studies in the area of vehicle routing the reader is refer to Jaillet and Wagner (2008). As already stated, the main problem studied in this field is the *online dial-a-ride problem* (OlDARP). In this problem, a request has a load of one unit and consists of a source point and a destination point in a metric space. A server (vehicle) with capacity Q starting at an origin point *serves* a request by performing the pickup at the source point and then delivering it at the destination point. Requests arrive online while the server is traveling, and they can be served at any time after the disclosure date (also called the release date). The objective consists of minimizing some function over the time at which deliveries are done.

Feuerstein and Stougie (2001) have studied the OlDARP with two different objectives: the minimization of the time the last destination is served (makespan) and the minimization of the sum of completion times (latency). For the OlDARP with the makespan minimization objective, the authors have first shown that any deterministic algorithm must have a competitive ratio of at least 2, independently of the server capacity Q. They later presented a simple 2.5-competitive algorithm called DLT (for Don't Listen while Traveling), which works as follows: once the server is at the origin at time t, it follows an optimal tour which serves all the requests not yet served (and received up to time t) and it goes back to the origin. The authors have then presented a more sophisticated 2-competitive algorithm for the case where $Q = \infty$. For the OlDARP with the objective of minimizing the latency, the authors have proved that any algorithm must have a competitive ratio of at least $1 + \sqrt{2}$.

Three online algorithms (REPLAN, IGNORE, and SMARTSTART) were proposed by Ascheuer et al. (2000) for the OlDARP in which $Q = 1$ and the objective is to minimize the makespan. In the algorithm REPLAN, when a new request is received, the server completes the carried request (if there is one), and then replans an optimal tour considering all the yet unserved requests. The authors have proved that REPLAN is 2.5-competitive. The algorithm IGNORE is the same as the DLT presented in the last paragraph and was independently proved to be 2.5-competitive. Finally, the authors have shown that the more sophisticated algorithm SMARTSTART is 2-competitive. The algorithm SMARTSTART, which was extended to handle a server with any capacity, is thus optimal since it closes the gap with respect to the lower bound.

Hauptmeier et al. (2000) have studied the online algorithms REPLAN and IGNORE in a continuously operating environment, i.e., an environment in which the time horizon is not bounded and where requests arrive indefinitely. Clearly, the objective of minimizing the makespan or the latency are inadequate since both will always be infinite. Thus, they have considered the objectives of maximal and average flow time. The flow time of a request under a solution s is the difference between its completion time using solution s and its release time. The authors have then shown that, under a clearly defined reasonable load restriction, there is a bound for the maximal and average flow time in the algorithm IGNORE, but no bound exists in the case of the algorithm REPLAN. This result is in contrast with the property proved by Ascheuer et al. (2000) that both algorithms have the same competitive ratio for the OlDARP.

Lipmann et al. (2004) have studied a modified version of the OlDARP called the *OlDARP under incomplete ride information* in which at the release date of a request, only the pickup point is revealed. The information about the delivery point is available when the pickup operation is performed. There are some applications in which this model is more realistic than the OlDARP such as, for example, the problem of scheduling an elevator. For the case in which preemption is allowed (i.e., the server is allowed to preempt any ride at any point and proceed later) the authors have proved that any deterministic algorithm for their problem must have a competitive ratio of at least 3. In addition, they have presented an algorithm called SNIFFER which preempts rides only at the pickup point (just to learn the delivery point) which is 3-competitive and thus optimal. For the case in which the server is not allowed to preempt, different lower bounds on the competitive ratio of any deterministic algorithm were proved, depending on the server capacity.

5.2. Heuristics

One of the few analytical studies for a dynamic and stochastic VRPPD was carried out by Swihart and Papastavrou (1999). The authors considered the problem of routing a single vehicle to serve incoming transportation requests arriving according to a Poisson process. Pickup and delivery locations are independent and uniformly distributed over a convex region with an Euclidean metric. The objective is to minimize the expected time during which requests remain in the system. Bounds on the performance of some proposed routing policies are presented both for the unit-capacity vehicle and for the multi-capacity vehicle. In the same vein, Waisanen et al. (2008) have studied two versions of the uncapacitated multiple vehicle case of this problem. In the first version, when a request arrives only the pickup location is known and the delivery location is received when the vehicle performs the pickup. In the second version, which is the most common, the pickup and the delivery locations are available when the request is received. For both versions, the authors have derived a lower bound on the objective for any control policy and have presented a control policy achieving the lower bound within a constant factor. Waisanen et al. (submitted for publication) have extended some of these results for dynamic *multi-stage* vehicle routing problems in which a request requires service at more than two locations.

An optimization based algorithm for a real-world dynamic multi vehicle VRPPD was proposed by Savelsbergh and Sol (1998). It captures standard constraints as well as problem specific features such as vehicles with special characteristics and different capacities, time windows, and working period restrictions. The optimization has to be performed over an horizon of several days and the objective is to minimize the driver's pay which depends on many factors such as the distance traveled and the number of nights away from home. The authors decompose the problem into a series of static problems with the a subset of the known requests over a rolling-horizon framework. Each static problem is solved by means of a branch-and-price based heuristic. The objective function for the static problem was modified to improve the overall performance of the algorithm in the dynamic environment. The algorithm was tested over a 10-day real-life instance with more than 200 active requests at any time. When there were large numbers of requests, the solution given by the algorithm outperformed the one produced by planners.

A rolling-horizon based algorithm for a dynamic uncapacitated VRPPD with time windows was developed by Mitrović-Minić and Laporte (2004). The problem studied by these authors was motivated by the routing and scheduling problem faced by courier companies making pickup and delivery of letters and small parcels in the same day. The objective is to serve all requests while minimizing the total distance traveled. Each of the static problems solved is optimized with a two-phase heuristic. The first phase consists of a cheapest insertion procedure while the second one consists of a tabu search algorithm similar to that of Gendreau et al. (2006).

Although no probability distribution about incoming requests is considered, different waiting strategies were developed. Through a series of computational tests, the authors showed that the total traveled distance was reduced when an advanced waiting strategy was used, instead of no waiting at all. Similar results about the benefits of waiting were analytically proved by Branke et al. (2005) for a simplified problem.

Mitrović-Minić et al. (2004) have extended the previous algorithm by incorporating a double-horizon objective. Routes are evaluated by a short-term objective of minimizing route length and a long term objective of having greater slack time to insert new requests. The basic idea is that it is not important to minimize the route length over a long horizon of time since it is expected that new requests will arrive and routes will have to be modified. This double-horizon approach performed better than the single horizon strategy on a series of computational tests carried out on instances based on data collected from two courier companies located in Vancouver, Canada.

Another tabu search algorithm for the same problem was proposed by Gendreau et al. (2006) and uses a neighborhood structure based on ejection chains. No stochastic information about incoming requests is assumed. An adaptive memory is used to dynamically maintain a set of good quality solutions consistent with the current vehicle trajectories. In order to quickly produce at least one solution of high quality when a new request arrives, the request is inserted into each solution of the adaptive memory through a cheapest insertion procedure. The best solution is selected after applying a fast local search procedure based on ejection chains. The adaptive memory is also updated upon service completion at a customer location. As long as there are no incoming requests and no services are completed, the tabu search heuristic keeps running in an attempt to improve the routes in the adaptive memory. In the tabu procedure, a master–slave parallelization scheme is used to speed up the search. The tabu search algorithm outperformed simpler insertion methods on a set of daily instances with up to 33 requests per hour.

Sáez et al. (2008) have developed an adaptive predictive control algorithm for a capacitated dynamic VRPPD, based on genetic algorithms and fuzzy clustering. In their problem, time windows are not present and the objective function takes into account the waiting and travel time for the requests, as well as the idle travel time of the vehicles. The authors have used a hybrid adaptive predictive control (HAPC) scheme based on state space variables to model the problem. The objective takes into account future requests through the use of probabilities calculated from historical data, using a fuzzy clustering method. The HAPC is optimized by means of a genetic algorithm. Computational tests on a 120-request instance under a heterogeneous distribution have shown that the algorithm performs up to 22% better when it takes into account future requests instead of the myopic strategy that only takes into account the received requests.

Recently, Ghiani et al. (2009) have proposed algorithms for an uncapacitated Dynamic VRPPD arising in same day courier services. In their model, each request i has associated a non-decreasing and convex function $f_i(t)$, which expresses the inconvenience cost of performing the delivery at time t. Following the ideas of Van Hentenryck and Bent (2004), the authors evaluate potential solutions using an objective function that takes future requests into account through sampling. To reduce the computation times, sampling is only done under a short-term horizon. In the first algorithm, called anticipatory insertion procedure, solutions incorporating the incoming request are constructed using a simple insertion method. A local search scheme based on simple moves is used for the second algorithm called anticipatory local search. In a series of computational tests with up to 600 requests, both algorithms have outperformed by 11–69% the corresponding reactive algorithm which does not use sampling. Results also showed a non-significant improvement of the anticipatory local search with respect to the anticipatory insertion procedure.

6. The dynamic dial-a-ride problem

In the Dynamic DARP, requests consist of users that need to be transported from an origin to a destination. Generally, these problems contain several constraints which control user inconvenience. Examples of such constraints are tight time windows and maximum ride times. The main application of the DARP is the transportation of handicapped and elderly people in cities. Models and algorithms for the static and the dynamic versions of the DARP were surveyed by Cordeau and Laporte (2007).

One of the first studies on the Dynamic DARP was carried out by Psaraftis (1980) who considered the single vehicle case. In this problem, users wish to be serviced as soon as possible and the objective is to minimize a weighted combination of total service time and user dissatisfaction. First, an exact $O(n^2 3^n)$ dynamic programming algorithm was presented for the static case. For the dynamic case, when a new request arrives the static instance is updated and reoptimized by fixing the partial route already done. Due to the computational complexity of the static algorithm, only small instances can be solved.

Madsen et al. (1995) have presented an algorithm for a real-life multi-vehicle dynamic DARP consisting of up to 300 daily requests for the transportation of elderly and handicapped people in Copenhagen. The problem had many constraints such as time windows, multi-dimensional capacity restrictions, customer priorities and a heterogeneous vehicle fleet. Many objectives taking into account user satisfaction and service costs were considered through the use of weight parameters. When a new request arrives, it is inserted in a current route using an efficient insertion algorithm based on that of Jaw et al. (1986). Computational results on real-life instances with up to 300 requests and 24 vehicles have shown that the algorithm was fast enough to be used in a dynamic context and that it is capable of producing good quality solutions.

Horn (2002) have developed a software for demand-responsive passenger services such as taxis and variable route buses. His problem includes time window restrictions for the dynamic requests, capacity constraints and booking cancellations. The insertion of new requests is done through a minimum cost insertion scheme which is supplemented by a local search procedure executed periodically. The author has developed a heuristic for strategically relocating the idle vehicles, taking into account future request patterns. Computational experiments were carried out based on data from a taxi company in Gold Coast, Australia. Results from the tested instances, which had around 4200 requests and 220 taxis in a 24 h period, have shown that the proposed software performs well on dynamic instances of large size.

A parallel algorithm for the Dynamic DARP was developed by Attanasio et al. (2004). At the start of the planning horizon, a static solution is constructed based on the known requests using a tabu search algorithm. When a new request arrives, each of the parallel threads inserts the request randomly in the current solution and runs the tabu search for obtaining a feasible solution. If a feasible solution is found, the request is accepted and a post-optimization phase is executed. Computational results with different parallelization strategies are reported on instances with up to 144 requests.

An algorithm for a dynamic DARP was developed by Coslovich et al. (2006). In this problem, a driver might unexpectedly receive a trip demand by a person located at a stop and must decide quickly whether to accept it or not. The algorithm maintains a repository of feasible solutions that are compatible with the partial routes already performed. When a new request arrives, an efficient

insertion algorithm attempts to insert it in at least one of the solutions in the repository. The request is accepted only if the insertion algorithm succeeds. Once the new request is accepted, the repository of feasible solutions is updated. This procedure is executed while the vehicle is moving between two locations and therefore response time requirements are not tight. Capacity constraints are not considered. Computational results on artificial instances with up to 50 requests are reported.

Beaudry et al. (in press) have developed a two-phase algorithm for solving a complex dynamic DARP arising in the transportation of patients in hospitals. This problem contains several constraints such as requests with different degrees of urgency and equipment requirements, multiple modes of patient transportation, and soft time windows. New requests are inserted using a fast insertion scheme able to satisfy the real-time requirements. The second phase consists of a tabu search which attempts to improve the current solution. The algorithm succeeded in reducing the waiting times for patients while using fewer vehicles in computational experiments on real data from a German hospital.

Finally, Xiang et al. (2008) have studied a sophisticated Dynamic DARP in which travel and service times have a stochastic component. In addition, there may be no-shows or cancellations and vehicles may be stuck into traffic jams or breakdowns. New requests are inserted into the established routes by means of a local search procedure based on simple inter-trip moves. The local search was complemented with a diversification strategy that uses a secondary objective measuring the idle time of the vehicles. The algorithm was tested on several simulated instances and involving up to 610 requests.

Recently, Cordeau et al. (2007) have presented the *dial-a-flight problem* in which the objective is to model and optimize on-demand air charter services that are provided by around 3000 companies in the United States. In this problem, requests generally consist of multiple pickup and delivery pairs and strict rules over the service are imposed. The authors discuss these and other differences with respect to the DARP, and some solution approaches are presented.

7. Conclusions

The volume of research on dynamic pickup and delivery problems has increased significantly over the last decade. In particular, useful general concepts such as basic solution strategies, algorithmic performance assessment, and the use of information about future requests have been put forward in recent years. Dynamic PDPs are rooted in real-world applications and, thanks to faster algorithms and efficient real-time communication technologies, solutions can be now realistically implemented in practice.

There still remain, however, a number of interesting questions worthy of investigation. An example is the problem of computing an optimal waiting strategy (see Section 3.3) taking into account the request probability distribution. Another research topic concerns the modification of the objective function under a rolling-horizon algorithm, which can, in some cases, yield good results. We also expect to see in the near future more studies on policy analysis, and lower bounds for more complex dynamic and stochastic pickup and delivery problems. Finally, another potential area of research is the study of dynamic many-to-many and one-to-many-to-one PDPs which have not yet received significant attention.

Acknowledgements

This work was supported by the Canadian Natural Sciences and Engineering Research Council under grants 227837-04 and 39682-05. This support is gratefully acknowledged. Thanks are due to the referees for their valuable comments.

References

Anily, S., Hassin, R., 1992. The swapping problem. Networks 22, 419–433.
Ascheuer, N., Krumke, S.O., Rambau, J., 2000. Online dial-a-ride problems: Minimizing the completion time. In: STACS 2000. Lecture Notes in Computer Science, vol. 1770. Springer, Berlin, pp. 639–650.
Attanasio, A., Cordeau, J.-F., Ghiani, G., Laporte, G., 2004. Parallel tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. Parallel Computing 30, 377–387.
Beaudry, A., Laporte, G., Melo, T., Nickel, S., in press. Dynamic transportation of patients in hospitals. OR Spectrum.
Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., Laporte, G., 2007. Static pickup and delivery problems: A classification scheme and survey. Top 15, 1–31.
Borodin, A., El-Yaniv, R., 2005. Online Computation and Competitive Analysis. Cambridge University Press, Cambridge.
Branke, J., Middendorf, M., Noeth, G., Dessouky, M., 2005. Waiting strategies for dynamic vehicle routing. Transportation Science 39, 298–312.
Cordeau, J.-F., Laporte, G., 2007. The dial-a-ride problem: Models and algorithms. Annals of Operations Research 153, 29–46.
Cordeau, J.-F., Laporte, G., Potvin, J.-Y., Savelsbergh, M.W.P., 2007. Transportation on demand. In: Barnhart, C., Laporte, G. (Eds.), Transportation. Handbooks in Operations Research and Management Science, vol. 14. North-Holland, Amsterdam, pp. 429–466.
Coslovich, L., Pesenti, R., Ukovich, W., 2006. A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem. European Journal of Operational Research 175, 1605–1615.
Feuerstein, E., Stougie, L., 2001. On-line single-server dial-a-ride problems. Theoretical Computer Science 268, 91–105.
Gendreau, M., Guertin, F., Potvin, J.-Y., Séguin, R., 2006. Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. Transportation Research Part C 14, 157–174.
Ghiani, G., Guerriero, F., Laporte, G., Musmanno, R., 2003. Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies. European Journal of Operational Research 151, 1–11.
Ghiani, G., Manni, E., Quaranta, A., Triki, C., 2009. Anticipatory algorithms for same-day courier dispatching. Transportation Research Part E 45, 96–106.
Gutenschwager, K., Niklaus, C., Voß, S., 2004. Dispatching of an electric monorail system: Applying metaheuristics to an online pickup and delivery problem. Transportation Science 38, 434–446.
Hauptmeier, D., Krumke, S.O., Rambau, J., 2000. The online dial-a-ride problem under reasonable load. In: Proceedings of Algorithms and Complexity, Fourth Italian Conference. Lecture Notes in Computer Science, vol. 1767. Springer, Berlin, pp. 125–136.
Horn, M.E.T., 2002. Fleet scheduling and dispatching for demand-responsive passenger services. Transportation Research Part C 10, 35–63.
Hvattum, L.M., Løkketangen, A., Laporte, G., 2006. Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic. Transportation Science 40, 421–438.
Hvattum, L.M., Løkketangen, A., Laporte, G., 2007. A branch-and-regret heuristic for stochastic and dynamic vehicle routing problems. Networks 49, 330–340.
Ichoua, S., Gendreau, M., Potvin, J.-Y., 2000. Diversion issues in real-time vehicle dispatching. Transportation Science 34, 426–438.
Ichoua, S., Gendreau, M., Potvin, J.-Y., 2006. Exploiting knowledge about future demands for real-time vehicle dispatching. Transportation Science 40, 211–225.
Jaillet, P., Wagner, M.R., 2006. Online routing problems: Value of advanced information as improved competitive ratios. Transportation Science 40, 200–210.
Jaillet, P., Wagner, M.R., 2008. Online vehicle routing problems: A survey. In: Golden, B., Raghavan, S., Wasil, E. (Eds.), The Vehicle Routing Problem: Latest Advances and New Challenges. Springer, Berlin, pp. 221–237.
Jaw, J.J., Odoni, A.R., Psaraftis, H.N., Wilson, N.H.M., 1986. A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. Transportation Research Part B 20, 243–257.
Larsen, A., 2001. The dynamic vehicle routing problem. Ph.D. Thesis, Institute of Mathematical Modelling, Technical University of Denmark.
Lipmann, M., Lu, X., de Paepe, W.E., Sitters, R.A., Stougie, L., 2004. On-line dial-a-ride problems under restricted information model. Algorithmica 40, 319–329.
Lund, K., Madsen, O.B.G., Solomon, M.M., 1996. Vehicle routing problems with varying degrees of dynamism. Technical Report, Institute of Mathematical Modelling, Technical University of Denmark.
Madsen, O.B.G., Ravn, H.F., Rygaard, J.M., 1995. A heuristic algorithm for a dial-a-ride problem with time windows multiple capacities and multiple objectives. Annals of Operations Research 60, 193–208.
Mes, M., van der Heijden, M., van Harten, A., 2007. Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems. European Journal of Operational Research 181, 59–75.
Mitrović-Minić, S., Krishnamurti, R., Laporte, G., 2004. Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows. Transportation Research Part B 38, 669–685.
Mitrović-Minić, S., Laporte, G., 2004. Waiting strategies for the dynamic pickup and delivery problem with time windows. Transportation Research Part B 38, 635–655.

Powell, W.B., 1987. An operational planning model for the dynamic vehicle allocation problem with uncertain demands. Transportation Research Part B 21, 217–232.

Powell, W.B., Bouzanene-Ayari, B., Simpo, H.P., 2007. Dynamic models for freight transportation. In: Barnhart, C., Laporte, G. (Eds.), Handbooks in Operations Research and Management Science: Transportation, vol. 14. North Holland, pp. 285–365.

Powell, W.B., Sheffi, Y., Nickerson, K.S., Butterbaugh, K., Atherton, S., 1988. Maximizing profits for North American Van Lines' truckload division: A new framework for pricing and operations. Interfaces 18 (1), 21–41.

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

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

Pureza, V., Laporte, G., 2008. Waiting and buffering strategies for the dynamic pickup and delivery problem with time windows. INFOR 46, 165–175.

Regan, A.C., Mahmassani, H.S., Jaillet, P., 1998. Evaluation of dynamic fleet management systems. Transportation Research Record 1645, 176–184.

Sáez, D., Cortés, C., Núñez, A., 2008. Hybrid adaptive predictive control for the multi-vehicle dynamic pick-up and delivery problem based on genetic algorithms and fuzzy clustering. Computers and Operations Research 35, 3412–3438.

Savelsbergh, M.W.P., Sol, M., 1998. DRIVE: Dynamic routing of independent vehicles. Operations Research 46, 474–490.

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

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

Tjokroamidjojo, D., Kutanoglu, E., Taylor, G.D., 2006. Quantifying the value of advance load information in truckload trucking. Transportation Research Part E 42, 340–357.

Van Hentenryck, P., Bent, R.W., 2004. Online stochastic combinatorial optimization. Operations Research 52, 977–987.

Waisanen, H.A., Shah, D., Dahleh, M.A., 2008. A dynamic pickup and delivery problem in mobile networks under information constraints. IEEE Transactions on Automatic Control 53, 1419–1433.

Waisanen, H.A., Shah, D., Dahleh, M.A., submitted for publication. Fundamental performance limits for multi-stage vehicle routing problems.

Xiang, Z., Chu, C., Chen, H., 2008. The study of a dynamic dial-a-ride problem under time-dependent and stochastic environments. European Journal of Operational Research 185, 534–551.

Yang, J., Jaillet, P., Mahmassani, H.S., 1998. On-line algorithms for truck fleet assignment and scheduling under real-time information. Transportation Research Record 1667, 107–113.

Yang, J., Jaillet, P., Mahmassani, H.S., 2004. Real-time multivehicle truckload pickup and delivery problems. Transportation Science 38, 135–148.