Comparison of policies in dynamic routing problems

E. Angelelli[*], *N. Bianchessi*[*], *R. Mansini*[§], *M.G. Speranza*[*]

[*] *Dipartimento Metodi Quantitativi, University of Brescia, Italy*

[§] *Dipartimento di Elettronica per l'Automazione, University of Brescia, Italy*

December 4, 2008

Abstract

We consider a company that has to satisfy customers pick-up requests arriving over time every day. The overall objective of the company is to serve as many requests as possible at a minimum operational cost. When organizing its business the company has to fix some features of the service that may affect both service quality and operational costs. Some of these features concern the time a request is taken into account to plan its the service, the associated deadline and the way requests are managed when the system is overloaded. In this paper we analyze several policies that can be implemented by the management of a carrier company in a multi-period context. For example, a company might reject all the requests that cannot be feasibly scheduled or accept all the requests and rely on a backup service in order to serve requests that are difficult to handle. Another interesting issue considered in this paper is the impact of collaborative service where two or more carrier companies, with their own customers, decide to share customers in order to optimize the overall costs. We set up a general framework to allow comparison of alternative service policies. Extensive computational results evaluating the number of lost requests and the distance traveled provide interesting insights.

Keywords: Dynamic Multi-Period Routing Problems, Postponable Requests, Management Policies.

1 Introduction

Dynamic settings are receiving increasing attention in routing problems thanks also to a wider use of communication devices in vehicles equipment. Nowadays, the use of GPS systems allows a central unit to constantly know the location of vehicles and to take dynamic decisions on the basis of the overall situation of vehicles and customers. Such situation evolves during the day and previous plans may be modified because new requests are issued by customers or because some unexpected event took place such as a delay due to traffic congestion. It is expected that such dynamic management improves the competitiveness of a company, allowing a better service at a lower cost.

While the literature on static routing problems is wide, the literature on dynamic routing problems is limited, though it has consistently grown in the last years. Comprehensive surveys on dynamic problems can be found in (Psaraftis, 1995), (Psaraftis, 1998) and, more recently, in (Ghiani et al., 2003). Among the most relevant contributions in this domain we recall (Savelsbergh and Sol, 1998) and (Yang et al., 2004) for the management of a dynamic fleet of vehicles; (Gendreau et al., 1999) and (Ichoua et al., 2000) for real-time vehicle routing and dispatching problems in long-distance courier services; (Mitrović-Minić et al., 2004) and (Mitrović-Minić and Laporte, 2004) for the dynamic pick-up and delivery problem with time windows and (Madsen et al., 1995) for a dynamic dial-a-ride system characterized by multiple capacities and multiple objectives. Finally, (Angelelli et al., 2007a) and (Angelelli et al., 2007b) perform a competitive analysis for some policies in a simple dynamic multi-period setting.

The dynamic setting we consider in this paper is the Dynamic Multi-Period Routing Problem (DMPRP) introduced in (Angelelli et al., 2008b). The problem is characterized by pick-up requests arriving in real time to a central depot and was originally motivated by a problem in the domain of courier service management. In this context the size of parcels moved in a pick-up service is not relevant. For this reason, the fleet of vehicles available for the service is assumed to be uncapacitated. Every morning these vehicles leave the depot and have to return to the depot at the end of the day. Thanks to modern communication technology, the company knows the exact position of its vehicles at any time instant and is able to forecast their positions in the near future. The company can react to on-line requests and possibly modify the previous

traveling plans. The distinctive features of this dynamic problem with respect to those analyzed in the literature is that each request receives a deadline that falls at the end of the same day or of the day after. Thus, on each day of service, requests can be classified as *postponable* or *unpostponable*. If a request is unpostponable it has to be inserted in the currently traveled routes, on the contrary if it is postponable it can be served either today or postponed to the day after. Moreover, every day, requests can be either *off-line* when they are known in advance (i.e. they have been previously issued but not served yet) or *on-line* when they come over in real-time while the vehicles are traveling. The objective of the company is to maximize the number of served requests at a minimum operational cost.

The most common approach used in the literature to solve a dynamic problem is based on a repeated re-optimization of the off-line problem. In (Angelelli et al., 2008b) and (Angelelli et al., 2008a), the authors introduce different *short term strategies* characterized by a look-ahead period and a short term objective. A re-optimization problem is then defined and iteratively solved by means of a Variable Neighborhood Search (VNS) meta-heuristic. An extensive computational analysis of the impact each short term strategy has on the long term objective of the problem is provided.

In the present paper, we consider the service policy assumed in (Angelelli et al., 2008b), which we call *dynamic*, and a number of policies arising in real cases. The performance that can be obtained by these policies is evaluated and compared to the performance that can be obtained by the *dynamic* policy. The effectiveness of each policy is evaluated by a proper implementation of the solution framework presented in (Angelelli et al., 2008b). Our aim is to point out possible improvements that a company can achieve in terms of number of served requests and operational costs. In this perspective we perform three different types of analysis, each one characterized by the application of alternative management policies. In the first analysis, we compare policies that differ about the management of requests that cannot be handled directly by the company. In one case (*dynamic*) no request is ever rejected and requests that cannot be served by the company are forwarded to a backup service at a high cost for the company. In the other case, requests that cannot be served are rejected. Of course in this case it is of crucial importance to decide as soon as possible whether a request should be accepted or rejected. The decision cannot be changed later on. In

the second analysis, we compare the *dynamic* policy where each new request is taken into account as soon as it is issued to a policy where all the requests issued during a day are analyzed at the end of the day and scheduled for the day after. Finally, we compare the *dynamic* policy when applied to a situation where fleets of vehicles are coordinated to serve a set of customers to the situation where separate fleets are run independently on pre-assigned sub-sets of customers. This comparison may be of interest to independent companies that want to understand the advantages of collaborating and centralizing the management of their individual fleets. Comparisons are made by means of extensive computational tests. Results are evaluated by the number of lost requests and the total distance traveled.

The paper is organized as follows. In Section 2 we present the dynamic context in which the carrier company operates and the solution framework by means of which we implemented all the policies analyzed in this paper. In Section 3 we describe the policies we took into consideration for comparison and how we used the framework to implement them. In Section 4 we describe the computational results of the analysis. Finally, we draw some conclusions.

2 The Dynamic Multi-Period Problem

A fleet of uncapacitated vehicles $V = \{v_1, \ldots, v_m\}$ is available to satisfy requests issued by customers. The positions of the vehicles are known to the central depot at any time during the day. Moreover, the vehicles can communicate with the central depot. At the beginning of each day a set of requests are known that have to be served during the day (*unpostponable* requests). These requests are assigned to the vehicles and the vehicles leave the depot and start traveling on the basis of an initial plan. During the day new requests may be issued by customers. Unpostponable requests can be accepted only until a fixed time L in the morning (e.g. noon or 1:00 pm). All the requests issued during the day that can be served in the same day or postponed to the day after are defined as *postponable*. The time length of each working day is equal to τ. This is also the maximum time available to each vehicle route, i.e. we will refer to the length of a route by meaning a time length. Decisions are repeated over a time horizon of T days.

All requests require a pick-up service. In fact, it is assumed that delivery requests are not consistent with this dynamic setting, since if a delivery request is issued during the day, then a vehicle cannot be deviated to serve the new customer. Moreover, if a vehicle leaves the depot with the load to be delivered to a customer, the service of that customer cannot be later assigned to a different vehicle. In case the company has to face both pick-up and delivery requests, the assumption is that the fleet is divided into two parts, a part dedicated to the delivery service and the other part dedicated to the pick-up service. The part dedicated to the delivery service works as traditionally in a static context where the vehicles follow during the day the plan assigned to them at the beginning of the day. The part dedicated to the pick-up service is managed dynamically.

The central depot may elaborate new plans during the day and communicate the changes to the vehicles. The changes in a vehicle plan may concern the inclusion of new customers, the deletion of customers or both. The vehicle may receive the new plans at any time and possibly deviate from its previous route while traveling between two customers. The goal is the minimization of the total service cost over the whole horizon. Such major target has been formalized through two hierarchical objectives. The first one is the maximization of the number of requests directly served by the company, which is equivalent to the minimization of the number of not served requests, i.e. rejected or forwarded to the backup service, depending on the policy. The second one is the minimization of the length of the routes traveled.

2.1 The solution framework

In (Angelelli et al., 2008b) the authors introduced in a rolling horizon solution framework the concept of a *Short Term Strategy* (STS). A STS includes the definition of a re-optimization problem that is solved by means of a Variable Neighborhood Search (VNS) heuristic. Before the beginning of the day and then at regular intervals (re-optimization intervals) the re-optimization problem is solved. The first re-optimization problem considers unpostponable requests only and provides for each vehicle a route that starts and ends at the depot. The subsequent re-optimization problems take into account all known requests (postponable and unpostponable) and

provide for each vehicle a route that starts at the forecasted position of the vehicle at the end of the re-optimization according to the previously planned routes and ends at the depot.

Time is denoted during the day with $t \in [0, \tau]$. We indicate by $R_P(t)$ and $R_U(t)$ the set of postponable and unpostponable requests at a given time t, respectively. We also denote by $R(t) = R_P(t) \cup R_U(t)$ the total set of the requests known at time t. Let Δt be the length of the re-optimization interval and let $t' = t + \Delta t$. The set $R(t')$ differs from $R(t)$ for the inclusion of all the new requests which have become available during the last re-optimization interval Δt and for the elimination of all the requests served in the meantime.

A maximum time $OptTime \leq \Delta t$ is made available to the algorithm that solves each re-optimization problem. The solution found is implemented until the end of the next re-optimization phase. The generated routes are followed by the vehicles from time $t + OptTime$ to time $t' + OptTime$, that is until the routes obtained with the subsequent re-optimization become available.

2.2 The short term strategy

In (Angelelli et al., 2008b) several *Short Term Strategies* have been analyzed and compared. A *Short Term Strategy* (STS) consists of the following components:

1. *A look-ahead period*: The period of time over which the re-optimization problem is defined;

2. *A short term objective*: The criterion used to evaluate the quality of a solution in the re-optimization problem;

3. *A re-optimization problem*: The off-line problem which is formulated and solved, after a look-ahead period and a short term objective have been defined;

4. *A re-optimization interval*: The length of the time interval between the solution of two consecutive re-optimization problems.

Let r_P^1 and r_U^1 represent the number of postponable and unpostponable requests served today, respectively. Moreover, let r^1 and l^1 denote the total number of served

requests and the total length of the routes traveled in the current day, respectively. Let r^2 and l^2 denote the number of served requests and the total length of the routes traveled the day after, respectively. The class of strategies that in (Angelelli et al., 2008b) turned out to be the most successful has a *2-day look-ahead* period, and the following short term objective:

$$min \quad \alpha l^1 + (1-\alpha) l^2 + (r_P^1 + r^2) K_2 + r_U^1 K_1 \tag{1}$$

where α is a real number such that $0 \leq \alpha \leq 1$, and K_1, K_2 are negative constant values such that $K_1 \ll K_2 \ll 0$. The function maximizes the number of unpostponable served requests (term $r_U^1 K_1$) and, as second hierarchical objective, maximizes the total number of postponable requests to be served within the day after (term $(r_P^1 + r^2)K_2$). Actually, the requests that are postponable today will become unpostponable tomorrow and have to be served within tomorrow. Finally, the third hierarchical objective is the minimization of the weighted sum of the lengths of the routes traveled today and tomorrow (term $\alpha l^1 + (1-\alpha) l^2$). In this last term α is set to 1^- so that a decrease in the distance traveled today is to be preferred to any decrease in the distance traveled tomorrow.

Interested readers are referred to (Angelelli et al., 2008b) for more details on the analysis of strategies with a 2-day look-ahead period against those with a 1-day look-ahead period. For sake of clarity, we recall here that, in all the tested instances, the best 1-day strategy has shown a weaker performance with respect to the best 2-day strategy. Indeed, in the best 2-day strategy the number of not served requests is halved and the distance traveled is on average reduced as well.

Throughout the paper we indicate by *2-day look-ahead(Δt)* the *2-day look-ahead* strategies with the short term objective function (1) and α set to 1^-.

3 Comparison of policies

In the following we define a set of different policies and assume that the company has the opportunity to evaluate the pros and cons of each alternative in order to implement the best one. Each of the analyzed management policies can be implemented by a proper implementation of the solution framework presented in (Angelelli et al., 2008b).

In particular, for all but one policies the implementation is straightforward as they are directly implied by a particular strategy which belong to the *2-day look-ahead(Δt)* strategies.

We call *dynamic* the service policy where requests arriving over time can be either unpostponable (to be served on the same day) or postponable to the day after. We assume that all requests are accepted and the company guarantees their service. Since the fleet may not be sufficient to satisfy all the issued requests, a contract with a backup service company has to be made in such a way that, in case of need, some of the requests will be served by the backup company. In this case, the backup company is informed on the customers to be served at the end of the morning. The *dynamic* policy allows the company to postpone the decision about the customers to serve directly and the customers to be served by the backup company. At the same time the company offers a high level of service, since no customer is ever rejected. The company's first objective is to forward to the backup company as less requests as possible (not served requests) and secondly to minimize the traveled distance. The *dynamic* policy can be implemented by means of a *2-day look-ahead(Δt)* strategy for any fixed value of the parameter Δt.

Dynamic vs. Accept/reject policies

The *dynamic* policy assumes that no request is ever rejected. All requests are accepted by the call center and the decision whether the request will be served by the company itself or by a backup service is delayed as much as possible. For organizational reasons a company may not be interested to rely on a backup service. In this case, there is no guarantee that all incoming requests can be served, and the company can accept only a subset of the requests, others must be rejected. Obviously, the decision about each request should be taken as soon as possible, and once a decision is taken, it cannot be changed later on. We consider two policies of such kind, which we call *accept/reject* and *fixed day accept/reject*. In both cases, when a new request is issued, the central unit of the company can immediately check whether it can be served by the available fleet of vehicles. Thus, a decision can be taken on the basis of the already accepted requests possibly modifying the current routing plan. The company accepts to serve

the request if it can be feasibly inserted in the current plan. Otherwise, the company rejects the request. Policy *accept/reject* guarantees that all accepted requests will be served within their deadlines. Policy *fixed day accept/reject* is more customer oriented and fixes, at time of acceptance, the day on which the request will be served. The policies are compared in terms of the number of not served requests and the traveled distance. For all the policies, the number of not served requests represents a cost. In the *dynamic* policy the company pays a high fee for each request forwarded to the backup service. In the *accept/reject* ones, each not served request is a lost income.

We implement *accept/reject* and *fixed day accept/reject* policies as follows. We consider a *2-day look-ahead(Δt)* strategy with a value of the parameter Δt small enough to allow the definition of a new re-optimization problem as soon as a new request is issued. Then, we modify the solution framework so that if an arriving request is feasibly inserted in the current routes by the first re-optimization, it is accepted and never excluded by future re-optimizations. Otherwise, the request is rejected and permanently discarded from the system. Finally, in case of *fixed day accept/reject* policy the service day assigned to a new request after the first re-optimization has to be kept fixed. To this purpose, we properly modified the definitions of the neighborhoods RELOCATE and EXCHANGE presented in (Angelelli et al., 2008b).

Dynamic vs. Static policy

The *dynamic* policy is attractive because it may reduce operational costs and guarantee a better service level. However, it also implies additional costs due to the technical devices needed for the communication and a different organization. Do the benefits compensate the costs? We compare such policy to a policy where the vehicles follow the route plans made available at the beginning of the day and based on the requests arrived the day before. Since for this policy all requests issued during a day are analyzed only at its end we call it *static* policy. In practice, this policy applies when vehicles are not equipped with a communication system which allows the central unit to know their locations at each time instant and to dynamically change their route plans. The *static* policy can be implemented by setting the re-optimization interval Δt equal to the working time τ.

Collaborative vs. Individual dynamic transportation policy

Traditionally, transportation companies have focused their attention on controlling and reducing their own costs to increase profitability. More recently, companies have started to explore the possibility to share information with other companies and to develop common transportation plans with further reduction of costs. A collaborative transportation policy might open up cost saving opportunities that are impossible to achieve with an internal company policy.

We compare a *collaborative transportation* policy where the route plans of a fleet of vehicles are designed by a unique decision maker who brings together all customer requests to a policy where the same vehicles are managed independently (*individual transportation* policy). In both cases the service is assumed to be dynamic. The situation refers to the real case where the service provided by a company with a large fleet of vehicles is compared to that provided by different smaller companies which globally own the same number of vehicles but whose route plans are managed independently. Both policies can be implemented by means of a *2-day look-ahead(Δt)* strategy for any fixed value of the parameter Δt.

4 Computational results

4.1 Testing environment

The computational analysis has been carried out on randomly generated instances. For all instances, the requests are uniformly distributed over an area of 100×100 km^2. In particular, each request coordinates are randomly selected among the customer coordinates in the sets r1 and r2 of the Solomon's instances for the Vehicle Routing Problem with Time Windows (see (Solomon, 1987)). In all the instances, the coordinates of the depot are those of the Solomon's instances. Requests arrive over a planning horizon of $T = 5$ days. Each day has a working time of $\tau = 10$ hours (from 8:00 AM to 6:00 PM). Requests are assumed to arrive over time according to a Poisson process with daily arrival rate λ. Requests issued before 1:00 PM are considered unpostponable with probability 1/3; otherwise, they are considered postponable with deadline at the end of the day after. The service is provided by means of a fleet

of 3 uncapacitated vehicles, each of them traveling at a constant speed of 25 km/h. In order to make all results comparable, an additional day is considered to complete the work of the not yet served requests. In fact, if not so, an improper advantage might be obtained by postponing as many requests as possible from day T to day $T+1$.

Each instance belongs to a scenario characterized by the value of parameter λ. We consider 5 scenarios with $\lambda = 100, 200, 300, 400, 500$ and for each scenario we generate 5 different instances. Actually, to test *collaborative transportation* versus *individual transportation* policy we used 10 instances from scenario $\lambda = 300$. Moreover, as specified below, some changes have been applied to data in order to produce suitable input for the analyzed policies. The objective of the analysis is to evaluate the number of not served requests and the distance traveled for each policy in order to discuss the advantages or disadvantages of their application.

All computational experiments have been carried out on a 1.5 GHz Intel Pentium IV machine with 512 MB of RAM.

4.2 Re-optimization time interval influence

In this section we briefly discuss the setting for the parameters Δt and $OptTime$.

In (Angelelli et al., 2008b) values of the re-optimization time interval Δt greater than or equal to 3600 seconds were tested. As high values of $OptTime$ would imply forecasting the vehicle position in the distant future and to implement the new solution only afterwards, the authors set the re-optimization time to $\frac{1}{12}\Delta t$. In order to possibly identify a better re-optimization time interval Δt for the *2-day look-ahead* strategies we tested the values 30, 150, 300, 600, 900, 1800, 3600 seconds. For small values of Δt, $OptTime = \frac{1}{12}\Delta t$ is not a reasonable time limit for the VNS meta-heuristic. Thus, to implement the short term strategies we have set the parameter $OptTime$ equal to $max\{\frac{1}{12}\Delta t,\ 30\}$ seconds.

The trend of the number of lost requests as a function of Δt is shown in Figure 1 where the results for each scenario depending on λ are provided. It is evident that, independently of the scenario, the minimum number of lost requests is found for $\Delta t = 30$ seconds. In particular, if we consider those scenarios where the number of lost requests is quite high (i.e. $\lambda \geq 300$) this number reduces on average by 50.66%

when moving from $\Delta t = 3600$ to $\Delta t = 30$ seconds.

In Figure 2 we plot the average distance traveled per served request as a function of Δt under the different scenarios. Again, independently of λ, the average distance traveled to serve a request tends to reduce when Δt shrinks: the lower the time between two consecutive re-optimizations the more efficient the transportation service.

These preliminary experiments have led us to the decision to set $\Delta t = 30$ for all the policies but the *static* one.

4.3 Dynamic vs. Accept/reject policies

In the following we discuss the computational results obtained when comparing the *dynamic* policy to the *accept/reject* and *fixed day accept/reject* policies.

Table 1 is divided into three parts, one for each of the analyzed policies. The first column in each part provides the average number of lost requests when the corresponding policy is applied under the 5 different scenarios. Each column gap_d measures the average percent increase in the number of not served requests of the analyzed policy with respect to the *dynamic* policy. Similarly, $gap_{a/r}$ measures the percentage increase of the number of lost requests of the *fixed day accept/reject* policy when compared to its basic version without fixed day. A negative percentage value (as for $\lambda = 200$) means that, on average, the fixed day policy has provided a lower number of lost requests. If we consider the scenarios with $\lambda \geq 300$ the *accept/reject* policy looses on average 71.91% more requests than the *dynamic* policy. The result is even worse if we consider the *fixed day accept/reject* policy.

Table 2 shows the average total distance traveled by the vehicles. The meaning of each column is the same as for Table 1. When considering the traveled distances, one can notice that, for each scenario, all policies tend to completely use the available vehicles. But for only one case ($\lambda = 300$), the average distances traveled by applying the three policies differ from each other for a percentage less than 1%. Thus, the conclusions can be drawn from the analysis of the number of not served requests only. Table 1 clearly shows that the *dynamic* policy largely dominates the *accept/reject* policy, whereas the *fixed day accept/reject* policy does not substantially worsen the performance of the *accept/reject* policy.

4.4 Dynamic vs. Static policy

In order to make a fair comparison between the *static* and the *dynamic* policy, we had to modify the test instances. The reason is that in the test instances there are unpostponable customers that typically arrive during the morning and have to be served before the end of the day. Such customers would certainly be lost by the *static* policy that optimizes the routes at the beginning of each day and does not revise the routes until the day after. Thus, we slightly modified the test instances by making the unpostponable customers that arrive during the morning postponable. Then, the *static* policy examines the customers waiting for service at the beginning of each day and optimizes the routes of the day. The customers that cannot be accommodated in those routes are lost. The *dynamic* policy has the advantage of reacting to the arrivals of customers during the day and serving a customer either during the arrival day or the day after. From the customer viewpoint the service is the same. Customer requests are served within the day after the arrival day.

In Table 3 for each scenario (value of λ in column 1) the average number of not served requests for the *dynamic* policy and for the *static* policy are shown. We observe a dramatic difference between the two policies. The *static* policy misses a number of requests about one order of magnitude higher than the *dynamic* policy. It is clear that a greater degree of flexibility in planning the service largely improves the results.

In Table 4 we report the average distance traveled per served request ($\bar{l}$) when the *dynamic* policy and the *static* policy are applied. The results confirm those of Table 3. The *static* policy performs worse than the *dynamic* policy. The percentage differences $gap(\%)$ between the distances, though remarkable, are not as large as for the number of not served requests. In conclusion, the *static* policy seems to be largely dominated by the *dynamic* policy both in terms of number of served requests and in terms of average distance traveled per served request.

4.5 Collaborative vs. Individual dynamic transportation policy

In this class of experiments we have selected the scenario characterized by $\lambda = 300$ and increased the number of test instances to 10.

We compare the performance of the *collaborative transportation* policy where a single company manages a 3-vehicle fleet and plans the service of the whole set of customers to the performance obtained by the *individual transportation* policy where customers are partitioned into three subsets and three different companies run a 1-vehicle fleet each serving one subset each. Instances for the *individual transportation* policy are generated from instances for the *collaborative transportation* policy by partitioning the set of requests in three sets. Each subset is an independent instance to be solved by a 1-vehicle fleet. Obviously, the partition of the set of requests can be done in many different ways. We considered two different cases. We first assumed that the three companies have customers who are uniformly distributed over the same area (case *a*). Then, we assumed that the three companies serve disjoint zones. In this case we partitioned the area as depicted in Figure 3 (case *b*). In case *a*, the three small instances are generated by considering one request every three requests starting with the first, the second and the third request of the original instance, respectively. Since, in each instance, the requests are sorted in increasing order of their release time, the requests in each small instance result not only geographically uniformly distributed, but uniformly distributed over time. In case *b* the instance is partitioned into three smaller instances according to the geographical position of requests.

For each case two tables are reported. The first type of table (Tables 5 and 7) compares, for each instance, the number of not served requests by the *collaborative transportation* policy (column #) to the number of not served requests by each single vehicle (columns $\#_i$) by the *individual transportation* policy. The last two columns ($\sum_{i=1}^{i=3} \#_i$) and (gap(%)) report the total number of not served requests by the *individual transportation* policy and the percentage difference with respect to the corresponding value obtained by the *collaborative transportation* policy. The last line provides average values taken over the 10 instances. The second type of table (Tables 6 and 8) has a similar structure, but reports results in terms of traveled distance.

In case *a* we see that by moving from individual transportation systems to a collaborative one the number of lost requests decreases, on average, by an order of magnitude (see Table 5). The dramatic advantages provided by a *collaborative transportation* policy are evident also in terms of traveled distance. The total distance traveled by the three vehicles managed individually is, on average, higher by 4.77%

even though the number of served requests is much lower (see column 5 in Table 5).

In case *b* the performance of the *individual transportation* policy improves a lot with respect to case *a*. As shown in Tables 7 and 8, the number of not served requests in *individual transportation* is much higher than in *collaborative transportation*. Besides, the total traveled distance is, but for two instances, larger for *individual transportation*.

Conclusions

In this paper we analyzed, from a management point of view, a dynamic environment for a carrier company. The vehicles are equipped with communication devices that make it possible to a central unit to evaluate in real-time new service requests and re-route the vehicles whenever beneficial. We tested different scenarios where each scenario is characterized by a different requests arrival rate. The first result we obtained is that a reduction of the interval between two consecutive re-optimizations of the service from 1 hour down to 30 seconds reduces the number of lost customers and the distance traveled. Thus, in all the subsequent experiments we have fixed the re-optimization interval to 30 seconds.

We then studied a number of different management policies a carrier may decide to follow to carry out the service to its customers. We analyzed the policies by evaluating two performance criteria: the number of lost customers and the distance traveled by the vehicles.

One of the management issues a carrier has to face is whether to give an immediate accept/reject answer to a service request on the basis of the previously accepted requests and the fleet of available vehicles or to accept all customers and, in case of need, to make use of a back-up company at a later time. The results of the experiments have shown that the latter policy is much more effective as the number of customer served by the vehicles of the carrier increases on average by more than 70% whereas the total distance traveled increases only slightly.

It is well known that a large carrier can take advantage of its size to increase the average load of a vehicle and reduce the number of empty trips with respect to a smaller carrier. Small carriers are frequent in Europe and in Italy in particular.

Small carriers may merge or at least implement a collaborative policy to improve their overall performance. The comparison of the behavior of three carriers that own one vehicle each to the behavior of a hypothetical carrier that owns the three vehicles shows that a collaborative policy dramatically increases the number of served customers and at the same time reduces the traveled distance.

Finally, we compared a dynamic policy with a static one. The dynamic policy requires investment costs in communication devices and a more complex and dynamic organization. Is it worthwhile? We have shown that the dynamic policy reduces the number of lost customers by almost an order of magnitude while reducing at the same time the traveled distance.

While in most cases a model and a solution algorithm for a routing problem are designed and tested with an operational point of view, we have taken in this paper a managerial point of view and have quantified, thanks to the availability of a software for the optimization in a dynamic routing environment, the advantages and disadvantages of different management policies.

A final remark concerns the use of the proposed dynamic strategies in a real setting. In the experiments we discussed in this paper we focused on the case of daily constant values of the arrival rate λ. We tested different values of λ to evaluate the impact of heavy/light service conditions on the various management policies. Daily constant values of λ allowed us to make a clearer comparison of the management policies. In a real setting, one should expect fluctuations in λ. A time series analysis may reveal the actual requests pattern and a good forecasting model may be used to obtain data on which specific experiments could be run that would improve the decision making process in the real setting at hand.

References

E. Angelelli, N. Bianchessi, R. Mansini and M.G. Speranza. "Management Policies in a Dynamic Multi-Period Routing Problem." In L. Bertazzi, J. van Nunen and M.G. Speranza, editors, *Innovation in distribution logistics*, Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, to appear (2008a).

E. Angelelli, N. Bianchessi, R. Mansini and M.G. Speranza. "Short Term Strategies for a Dynamic Multi-Period Routing Problem." *Transportation Research: Part C*, **to appear** (2008b).

E. Angelelli, M.W.P. Savelsbergh and M.G. Speranza. "Competitive Analysis for Dynamic Multi-Period Uncapacitated Routing Problems." *Networks*, **49**:308–317 (2007a).

E. Angelelli, M.W.P. Savelsbergh and M.G. Speranza. "Competitive Analysis of a Dispatch Policy for a Dynamic Multi-Period Routing Problem." *Operations Research Letters*, **35**:713–721 (2007b).

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

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

S. Ichoua, M.Gendreau and J-Y Potvin. "Diversion Issues in Real-Time Vehicle Dispatching." *Transportation Science*, **34**:426–438 (2000).

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

S. Mitrović-Minić, R. Krishnamurti and G. Laporte. "The double-horizon heuristic for the dynamic pickup and delivery problem with time windows." *Transportation Research Part B*, **38**:669–685 (2004).

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

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

H.N. Psaraftis. "Dynamic vehicle routing problems." In B. L. Golden and A. A. Assad, editors, *Vehicle Routing: Methods and Studies*, pages 223–248. Elsevier Science, Amsterdam, 1998.

M.W.P. Savelsbergh and M. Sol. "Drive: Dynamic routing of independent vehicles." *Operations Research*, **46**:474–490 (1998).

M.M. Solomon. "Algorithms for vehicle routing and scheduling problems with time window constraints." *Operations Research*, **35**:254–266 (1987).

J. Yang, P. Jaillet and H. Mahmassani. "Real-Time Multivehicle Truckload Pickup and Delivery Problems." *Transportation Science*, **38**:135–148 (2004).

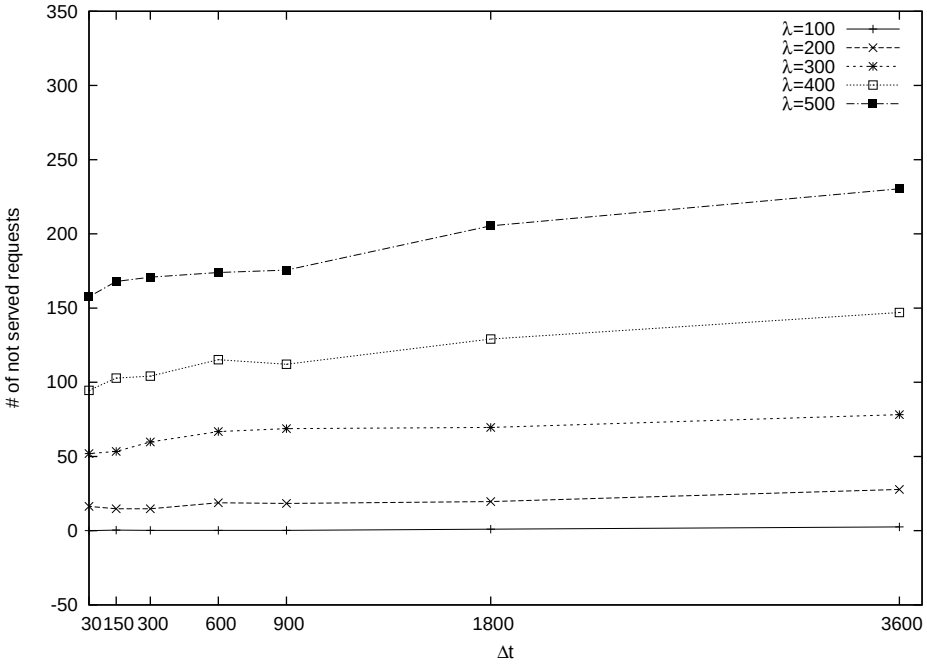

Figure 1: *2-day look-ahead* strategies with different Δt: number of not served requests.

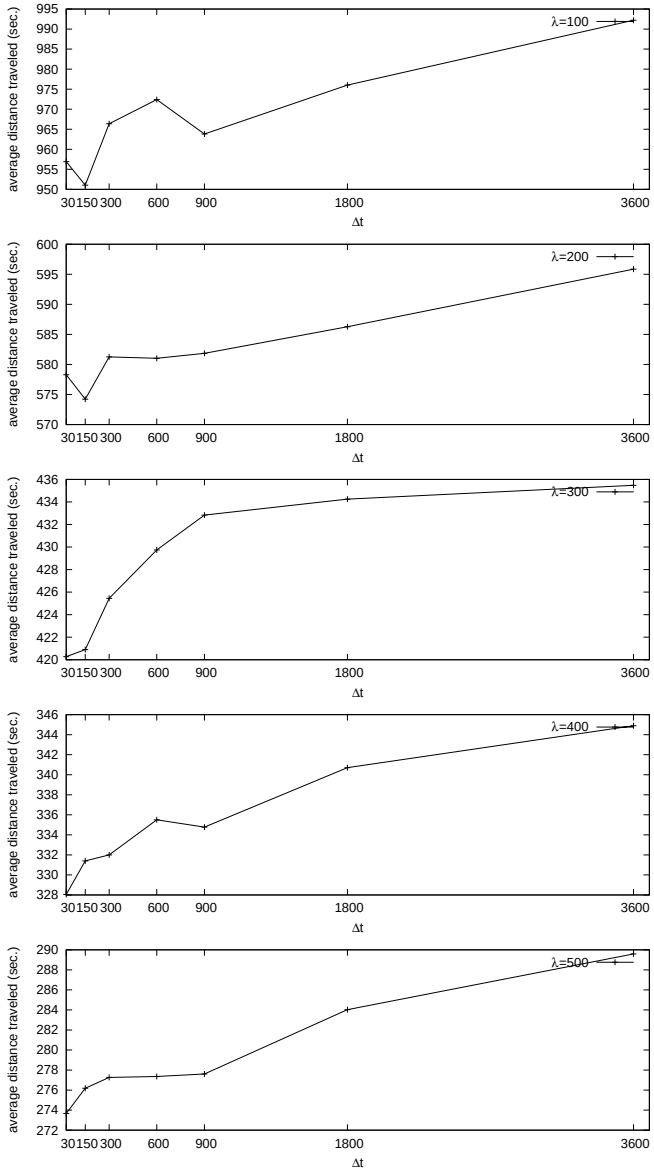

Figure 2: *2-day look-ahead* strategies with different Δt: average distance traveled per served request.

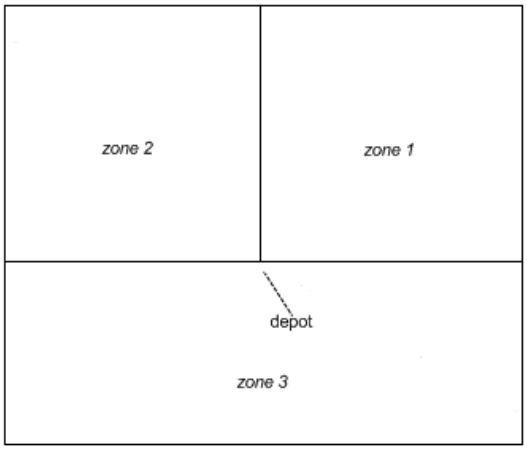

Figure 3: A partition of the initial geographical area.

	dynamic	***accept/reject***		***fixed day accept/reject***		
λ	#	#	$\text{gap}_d(\%)$	#	$\text{gap}_d(\%)$	$\text{gap}_{a/r}(\%)$
100	0.00	0.00	-	0.40	100.00	100.00
200	16.40	21.00	28.05	19.00	15.85	-9.52
300	52.00	88.40	70.00	101.60	95.38	14.93
400	94.60	162.60	71.88	175.80	85.84	8.12
500	157.6	274.00	73.86	276.60	75.51	0.95

Table 1: *dynamic* vs. *accept/reject* policies: average number of not served requests.

	dynamic	***accept/reject***		***fixed day accept/reject***		
λ	l	l	$\text{gap}_d(\%)$	l	$\text{gap}_d(\%)$	$\text{gap}_{a/r}(\%)$
100	513482.63	511915.97	-0.31	516256.46	0.54	0.85
200	588000.22	588730.57	0.12	587590.58	-0.07	-0.19
300	615194.60	618585.07	0.55	621880.01	1.09	0.53
400	627632.89	632123.91	0.72	633111.51	0.87	0.16
500	633372.85	636490.40	0.49	636841.83	0.55	0.06

Table 2: *dynamic* vs. *accept/reject* policies: average distance traveled.

Scenario	*dynamic*	*static*
λ	#	#
100	0.00	1.40
200	3.20	62.80
300	19.40	193.60
400	38.20	329.40
500	79.60	501.80

Table 3: *dynamic* vs. *static* policy: average number of not served requests.

Scenario	*dynamic*	*static*	
λ	$\bar{l}$	$\bar{l}$	$gap(\%)$
100	799.14	898.10	12.38
200	494.62	536.41	8.45
300	366.59	388.61	6.01
400	283.00	301.36	6.49
500	232.44	251.30	8.12

Table 4: *dynamic* vs. *static* policy: average distance traveled per served request.

instance	#	$\#_1$	$\#_2$	$\#_3$	$\sum_{i=1}^{i=3} \#_i$
1	46	184	183	168	535.00
2	54	192	185	198	575.00
3	44	203	195	193	591.00
4	54	183	188	193	564.00
5	62	192	183	198	573.00
6	53	194	194	185	573.00
7	46	191	187	175	553.00
8	44	187	180	181	548.00
9	39	183	183	187	553.00
10	66	205	199	193	597.00
	50.80	**191.40**	**187.70**	**187.10**	**566.20**

Table 5: *collaborative* vs. *individual* dynamic transportation policy: number of not served requests (case a).

instance	l	l_1	l_2	l_3	$\sum_{i=1}^{i=3} l_i$	gap (%)
1	611830.00	213116.62	213115.80	213234.55	639466.97	4.52
2	619250.07	214464.66	214033.89	213555.14	642053.69	3.68
3	617810.59	215376.95	213560.64	214432.88	643370.47	4.14
4	607299.61	214673.09	212627.13	213197.80	640498.02	5.47
5	619782.75	214658.65	213588.85	214084.49	642331.99	3.64
6	595790.27	214671.30	213490.68	209136.81	637298.79	6.97
7	609354.19	214887.06	214401.58	214989.29	644277.93	5.73
8	615016.16	214169.54	212961.75	214310.70	641441.99	4.30
9	614192.44	214833.80	214613.17	214843.07	644290.04	4.90
10	613412.14	214297.79	212961.39	213079.88	640339.06	4.39
	612373.82	**214514.95**	**213535.49**	**213486.46**	**641536.90**	**4.77**

Table 6: *collaborative* vs. *individual* dynamic transportation policy: distance traveled (case a).

instance	#	$\#_1$	$\#_2$	$\#_3$	$\sum_{i=1}^{i=3} \#_i$	gap (%)
1	46	27	3	48	78	69.57
2	54	22	3	58	83	53.7
3	44	26	2	52	80	81.82
4	54	23	3	46	72	33.33
5	62	26	3	53	82	32.26
6	53	27	6	47	80	50.94
7	46	16	2	48	66	43.48
8	44	17	4	47	68	54.55
9	39	19	3	47	69	76.92
10	66	24	4	69	97	46.97
	50.80	**22.70**	**3.30**	**51.50**	**77.50**	**54.35**

Table 7: *collaborative* vs. *individual* dynamic transportation policy: number of not served requests (case b).

instance	l	l_1	l_2	l_3	$\sum_{i=1}^{i=3} l_i$	gap (%)
1	611830.00	213116.62	188586.55	213589.85	615293.02	0.57
2	619250.07	214464.66	195477.83	210854.26	620796.75	0.25
3	617810.59	215376.95	189799.97	214023.75	619200.67	0.23
4	607299.61	214673.09	199177.22	211027.24	624877.55	2.89
5	619782.75	214658.65	191411.73	211735.59	617805.97	-0.32
6	595790.27	214671.30	190108.76	212366.33	617146.39	3.58
7	609354.19	214887.06	194091.55	210730.19	619708.80	1.7
8	615016.16	214169.54	189295.37	208000.56	611465.47	-0.58
9	614192.44	214833.80	196213.97	211233.75	622281.52	1.32
10	613412.14	214297.79	195518.58	210495.04	620311.41	1.12
	612373.82	**214514.95**	**192968.15**	**211405.66**	**618888.76**	**1.08**

Table 8: *collaborative* vs. *individual* dynamic transportation policy: distance traveled (case b).

List of Figures

List of Tables