Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

SEN

Software Engineering

Software ENgineering

Dynamic Routing Problems with Fruitful Regions: Models and Evolutionary Computation

J.I. van Hemert, J.A. La Poutré

REPORT SEN-E0410 AUGUST 2004

CWI is the National Research Institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organization for Scientific Research (NWO).
CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

ISSN 1386-369X

Dynamic Routing Problems with Fruitful Regions: Models and Evolutionary Computation

ABSTRACT
We introduce the concept of fruitful regions in a dynamic routing context: regions that have a high potential of generating loads to be transported. The objective is to maximise the number of loads transported, while keeping to capacity and time constraints. Loads arrive while the problem is being solved, which makes it a real-time routing problem. The solver is a self-adaptive evolutionary algorithm that ensures feasible solutions at all times. We investigate under what conditions the exploration of fruitful regions improves the effectiveness of the evolutionary algorithm.

2000 Mathematics Subject Classification: 90C27
1998 ACM Computing Classification System: I.2.8, F.2.2
Keywords and Phrases: dynamic vehicle routing;fruitful regions;evolutionary computation
Note: This work is part of DEAL (Distributed Engine for Advanced Logistics) supported as project EETK01141 under the Dutch EET programme.

Dynamic Routing Problems with Fruitful Regions: Models and Evolutionary Computation*

J.I. van Hemert and J.A. La Poutré

Dutch National Research Institute for Mathematics and Computer Science,
P.O. Box 94079, NL-1090 GB Amsterdam, The Netherlands
{jvhemert,hlp}@cwi.nl

Abstract. We introduce the concept of fruitful regions in a dynamic routing context: regions that have a high potential of generating loads to be transported. The objective is to maximise the number of loads transported, while keeping to capacity and time constraints. Loads arrive while the problem is being solved, which makes it a real-time routing problem. The solver is a self-adaptive evolutionary algorithm that ensures feasible solutions at all times. We investigate under what conditions the exploration of fruitful regions improves the effectiveness of the evolutionary algorithm.

1 Introduction

Vehicle routing forms an interesting line of research, because of its high complexity and intractable nature, as well as because of its importance to the transport industry. It has been studied in many different forms, whereby introducing concepts as uncertainty [1] and pickup-and-delivery scenarios [2]. In practice, dynamic routing problems occur widely [3], in which matters are further complicated as one needs to cope with a lack of knowledge while solving.

We concentrate our study on dynamic vehicle routing problems over an extended time period. This paper reports on the problem of collecting loads and delivering them to a central depot, during which requested pickups appear dynamically, i.e., while vehicles are on-route. We consider having knowledge about the probability where loads appear, especially in the form of regions. This knowledge could have been obtained from, for instance, data of previous routes on customer demands. We address the problem whether this knowledge may improve upon the quality of solutions.

In this paper we shall provide a mathematical model of this dynamic routing problem, for which we will introduce an evolutionary algorithm in order to solve it. Our main contribution is the discovery that exploration of fruitful regions is beneficial to the effectiveness of the solver for various types of problem settings.

2 Relation to and embedding into previous research

Often in practice, a problem may develop over time, that is, some variables in the problem may change or be set at a later stage such that a new evaluation of the

* This work is part of DEAL (Distributed Engine for Advanced Logistics) supported as project EETK01141 under the Dutch EET programme.

problem is required. Basically, this inserts uncertainties and lack of information into the problem. The literature distinguishes between two ways of analysing such problems, stochastic routing and dynamic routing [4]. The latter is also referred to as real-time routing or on-line routing [5]. In stochastic routing we are presented with a routing problem that has a number of its variables stochastically determined. This still requires a solution to the static problem, i.e., an a priori solution, but we can use only an expected value of the optimisation function. Examples of stochastic variables are uncertain supplies [6] and stochastic travel times [7]. The dynamic routing problem requires that the solver keeps waiting for events to occur to which it should act accordingly. Such events may be more customer requests, but they can also be changed travel times or other incidents that endanger the success of the current plan. A solver should thus make sure that it creates its solution to the current status in such a way that it may be altered successfully to changes in the future. Our interest lies in the second approach as these become more common in practice, due to an increasing amount of computing and communication resources and due to an increasing amount of flexibility demanded by customers [3, 8].

In the same book [9] where Psaraftis [10] laid out the differences between dynamic and static vehicle routing problems, Powell [11] first reviews the idea of forecasting the uncertainties within dynamic routing. Powell's work is targeted towards solving assignment problems, where the objective is to assign available loads to individual vehicles such that the whole fleet of vehicles is able to maintain a steady flow of work. In [12], a model is presented that deals with long-haul transport, which has led to a hybrid model that assigns jobs to trucks and forecasts jobs in future time periods. Thus, Powell's work is essentially about work flows, and not about dynamic routing.

Our objective is to use the concept of demand forecasting and use this in dynamic routing problems that occur real-time, i.e., the vehicles are on route when the problem changes. This is different to the work of Laporte and Louveaux, which is concerned with creating routes beforehand. It is also different to Powell's work on forecasting, as he uses it in combination with the assignment problem in the application area of dispatching, a fact that is underlined in [13]. In the assignment problem, the routing aspect, i.e., the optimisation problem where one minimises route length or the number of vehicles, is fully ignored in favour of having the fleet spread over the network such that the expected loads are covered.

In this paper, the application lies in transport that occurs in and between areas with a dense distribution of customers, where routing is required to achieve cost effective solutions. The problem is a dynamic in a sense that customer requests are handled dynamically, and where the problem is solved in real-time. Furthermore, the solver possibly makes use of the knowledge about high-potential, i.e., fruitful, regions.

3 Definition of the problem

A routing problem consists of a set of nodes N, connected by a graph $G = (N, E)$ with edges E that are labelled with the cost to traverse them, a set of loads L and a set of vehicles V. An optimisation function determines the goal of the routing

problem, which is further specified by a set of constraints that places restrictions on the aforementioned resources. Here we consider loads that need to be picked up from nodes and then delivered to a central depot.

A load $l \in L$ is a quadruple $\langle s, d, t, \Delta \rangle$, where $s \in N$ is the source node, and $d \in \mathbb{N}$ is the size of the load, which is fixed to one here. $t \in T$ is the entry time, i.e., the time when the load is announced to the solver, with T the time period of the problem. Every load should be delivered to a central depot $n_0 \in N$ within Δ time steps with $l \in L$, i.e., before $t + \Delta$. Here we assume that Δ is equal for each load.

A vehicle $v \in V$ is characterised by a capacity q and a start position $sp \in N$. We assume that all vehicles have the same capacity. Each vehicle has a speed of 1 and starts at the depot ($sp = n_0$). The dynamic status of a vehicle v, i.e., its route, assigned loads and current position, is defined with two functions. Both functions use the assumption that a vehicle always travels directly from node to node, i.e., it is not possible to stop half way and choose another route. This enables us to talk about the position of a vehicle in terms of nodes. We define the current position of a vehicle v as $p \in N$ where p is the node last visited by v. Then, we define the current situation using v and p as,

$$\text{current}(v, p) = \langle \boldsymbol{r}_{\leq p}, A_{\leq p}, A_p \rangle, \tag{1}$$

where $\boldsymbol{r}_{\leq p} = (r_0, \ldots, r_j)$ is an ordered list of nodes with $r_i \in N$ for $0 \leq i \leq j$ and $r_0 = sp$, denoting the route of the vehicle through the network up until and including node r_j where $r_j = p$. $A_{\leq p}$ is the set of loads that were assigned and were collected and delivered to the central depot by v so far, and A_p is the set of loads that were assigned and were collected by v so far, but not yet delivered.

$$\text{future}(v, p) = \langle \boldsymbol{r}_{>p}, A_{>p} \rangle, \tag{2}$$

where $\boldsymbol{r}_{>p} = (r_{p+1}, \ldots, r_k)$ is an ordered list of nodes with $r_i \in N$, which denotes the route planned for vehicle v from the next location r_{p+1} onwards. $A_{>p}$ is the set of loads that are assigned to be collected by v in the future.

We define some additional functions that are helpful in our futher formulation of the problem. The function arrival-time(v, p) returns the time vehicle v arrives at node $r_p \in N$ as the p^{th} node of its route. The function loads(v, p) returns the set of all loads assigned to vehicle v; loads$(v, p) = A_{\leq p} \cup A_p \cup A_{>p}$.

The constraints of the routing problem are given next.

$$\forall v \in V, \forall p : \text{future}(v, p) = \langle (r_{p+1}, \ldots, r_k), A_{>p} \rangle \wedge p + 1 = k \Rightarrow A_{>p} = \emptyset \tag{3}$$

$$\forall v_1, v_2 \in V, \forall p_1, p_2 : v_1 \neq v_2 \Rightarrow \text{loads}(v_1, p_1) \cap \text{loads}(v_2, p_2) = \emptyset \tag{4}$$

$$\forall v \in V, \forall p : \forall \langle s, d, t, \Delta \rangle \in A_p \cup A_{<p} : \text{arrival-time}(v, p) >= t \tag{5}$$

$$\begin{aligned} \forall l = \langle s, d, t, \Delta \rangle \in L : \forall v \in V : \text{current}(v, p) = \\ \langle r_{\leq p}, A_{\leq p}, A_p \rangle \wedge \langle s, d, t, \Delta \rangle \in A_p \wedge \exists p' : r_{p'} = d \wedge p \leq p' \Rightarrow \\ \text{arrival-time}(v, p') \leq t + \Delta \end{aligned} \tag{6}$$

$$\forall \langle v \rangle \in V : \forall p : \text{current}(v, p) = \langle r_p, A_{\leq p}, A_p \rangle \Rightarrow \sum_{\langle s, d, t, \Delta \rangle \in A_p} d \leq q \tag{7}$$

The constraint (3) restricts vehicles from carrying loads at the end of their route. To enforce that a load is carried by at most one vehicle the constraint (4) is used. The notion of time appears in three places; in the definition of a load, where the *entry time* is set, in constraint (5), where we make sure loads are only carried after their entry time, and in constraint (6), where we make sure loads are delivered at their destination within a fixed time after entering the system. Finally, constraint (7) is introduced to have vehicles carry only as much as their capacity lets them.

Three differences with the classic static routing problem can be identified as in this dynamic model:

1. vehicles may drive routes where they pass through nodes without performing a loading or unloading action,
2. loads may become available while vehicles are already on route, and
3. it is let open that each load must be assigned to a vehicle.

The latter enables us to choose as the objective of the routing problem to carry as many loads as possible, i.e., to maximise the number of loads successfully delivered at the central depot: let p_v be the last position of vehicle v at the end of time period T, then $\max \sum_{v \in V} \text{loads}(v, p_v)$. In the experiments we shall report as a performance measure the success ratio, which is defined as the number of loads successfully delivered at the depot divided by the total number of loads generated during a run of the whole system.

4 Fruitful regions

In practice, customers are often clustered into regions as opposed to scattered around equally over the total domain [14]. Often, the amount of service requested in total from one region may differ much from another. This is due to all types of reasons, such as the production capacities of customer in a region. Depending on the distribution of the customers and the distribution of service requests, it might be beneficial for a vehicle to visit nodes in the customer graph that have a high probability for a service request in the near future.

The grouping of customers into clusters is achieved either a posteriori, by observing the distribution of service requests in historical data, or a priori, by modelling them explicitly. In both cases we assume that a partition $C = \{C_1, C_2, \ldots, C_c\}$ of clusters exists in the layout of the problem, where each $C_i \subseteq N$. Next, we define the vector $\boldsymbol{f} = (f_1, f_2, \ldots, f_{|N|})$, where f_j is the probability that a load in the simulation originates from customer $j \in N$. The following holds: $\sum_{j \in N} f_j = 1$. We define the potential of a cluster C_i as the sum over the probabilities of its nodes: $\sum_{j \in C_i} f_j$.

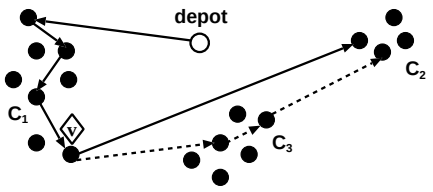


Fig. 1. An example of an anticipating route

An example of three clusters and one central depot is presented in Figure 1. The vehicle v that left the depot and then visited several nodes in C_1, is now on its route to a node in C_2. If the potential for new loads in C_3 is high enough, it might want to perform an anticipated move, visiting nodes in C_3 along the dotted lines, as this could provide another load without much additional driving.

5 Evolutionary algorithm

In the context of dynamic optimisation, evolutionary computation has the advantage of being able to cope with changes in the environment during its optimisation [15]. By maintaining a diverse set of candidate solutions, the evolutionary algorithm can adapt to the changes. Evolutionary algorithms can quickly converge to a suboptimal or optimal solution. Together with the current speed of processors, this makes it possible to keep providing solutions to a dynamic routing problem.

Bräysy and Gendreau [16] have reviewed many successful evolutionary algorithms and other meta-heuristic approaches for the classic vehicle routing problem. These were often tested using the well known Solomon benchmark [14]. All these implementations make extensive use of heuristics and other hybrid solutions to increase the efficiency and effectiveness of the final routing algorithm. Our objective is to verify whether or not the concept of fruitful regions can help increase performance. We shall start with a straightforward representation of the problem together with basic operators, then add the possibility to move vehicles to fruitful regions. Precise details of the evolutionary algorithm are given next.

5.1 An evolutionary algorithm for dynamic routing

Representation Each individual represents a plan for all the vehicles in the system. It is a list of routes, where each route corresponds to exactly one vehicle. A route consists of potential assignments for that vehicle, which it will perform in that order. Assignments are either pickup assignments or moves to a location. This representation is then decoded into a valid solution, i.e., a solution where none of the constraints are violated. Assignments that violate one or more time constraints are ignored by the decoder. When a vehicle reaches its capacity or when adding more assignments will violate a time constraint, the decoder forces a visit to the depot. Afterwards, this vehicle may be deployed to service customers again. This procedure is the same as used in [7]. The fitness of the individual will be based on the decoded solution. Although the decoding process may have a large impact on the fitness landscape, it is necessary as in a dynamic environment we must be able to produce valid solutions on demand.

Initialisation At the start, i.e., at $t = 0$, every available load $l \in \{\langle s, d, t, \Delta\rangle \in L|t = 0\}$ is randomly assigned to a vehicle. When fruitful regions are considered, a self-adaptative mechanism is used that lets an individual decide for itself how much it values to explore such regions. This employs an alpha value initialised uniform randomly between 0 and 1, which is further explained in the fitness function. The population size is set to 30.

Fitness Function The basis for the fitness function is the number of loads that can be transported successfully, i.e., that can be transported without violating the time and capacity constraints. We call these *feasible loads*. The fitness of an individual x is defined as,

$$\text{fitness}_1(x) = \frac{\#\text{ feasible loads}}{\#\text{ total loads available}}.$$

Only available loads are considered, that is, loads that are not yet picked up. The rationale here is that loads are removed from individuals of the evolutionary algorithm once they are picked up. As individuals only concern with future routing these loads play no role and should therefore not be included in the evaluation of an individual. Note that for assessing the overall quality of the evolutionary algorithm, we will use every load in the system over the whole period T.

We add the mechanism that allows the evolutionary algorithm to explore the possibility of employing vehicles to nodes that have not yet requested service. The fitness function, which has to be maximised, is extended with anticipated moves, i.e., moves that have no direct loading action attached. These moves are performed only when the constraints allow this. Thus, some moves may be cancelled during decoding to make sure that the final plan satisfies the constraints placed upon a vehicle because of previous assigned actions, such as, picking up a load. If we would perform such a move, the planning would become infeasible and hence, such a move is called an *infeasible anticipated move*. A candidate solution is penalised for every infeasible anticipated move in order to restrict the number of anticipated moves. The fitness of an candidate solution is decreased by the fraction of infeasible anticipated moves out of all planned anticipated moves. The amount with which this decreases the fitness function depends on an alpha value (α), which is encoded into every candidate solution. Using this self-adaptive mechanism, each candidate solution is able to determine by itself how much it values the effect of inserting anticipated moves.

$$\text{fitness}_2(x) = \text{fitness}_1(x) - \alpha\frac{\#\text{ infeasible anticipated moves}}{\#\text{ total anticipated moves of } x}.$$

To asses the effectiveness of the evolutionary algorithm, with or without anticipated moves, we measure the ratio of the loads successfully delivered to the total number of loads made available for transport, at the end of the run.

Selection A 2-tournament selection is used to select an individual for mutation. A generational scheme is used, known in evolution strategies as (μ, μ), with elitism of size one.

Variation operator Only mutation is considered. Two vehicles, possibly the same one, are chosen uniform randomly. In both vehicles two nodes are selected uniform randomly. If only one vehicle is chosen these nodes are chosen to be distinct. Then, the nodes are swapped. This way, visits, both for loading or as an anticipated move, can be exchanged between vehicles as well as within the route of one vehicle.

Furthermore, the individual's alpha value, which influences the fitness function, is changed according to the common update rule from evolution strategies.

Stop condition The algorithm is terminated once the time period T is expired, i.e., at the end of the simulation.

5.2 Dynamism

In practice, dealing with a dynamic problem means that we have limited time to contemplate the solution as vehicles need to be kept going and changes may be on their way. In reality we have the world and the solver running in parallel, and then we need to decide how often the solver's view of the world is updated. Here, we choose a time period based on the number of times the evolutionary algorithm evaluates one of its candidate solutions. These fitness evaluations take up most of the time in an evolutionary algorithm. For each time unit of the simulation the evolutionary algorithm may perform one generation. Thus the evolutionary algorithm will perform *population size* $\times$ *time steps* $(= 30|T|)$ fitness evaluations in a simulation run.

The whole simulation operates by alternatively running the evolutionary algorithm and the simulated routing problem. The routing simulator calculates when the next event will occur, e.g., a vehicle will pickup or deliver a load, or, a load is announced for pickup. Then, the evolutionary algorithm may run up until this event occurs. This way we simulate an interrupt of the evolutionary algorithm when it needs to adapt to changes in the real world. The best individual from the last generation before the "interrupt" is used to update the assignments of the vehicles in the routing simulation. Then, the routing problem is advanced up until the next event. Afterwards, the individuals of the evolutionary algorithm are updated by removing finished assignments and adding loads recently made available. This process is repeated for time period T. Note that it is possible that multiple events occur at the same time, which will be handled in one go.

6 Experiments

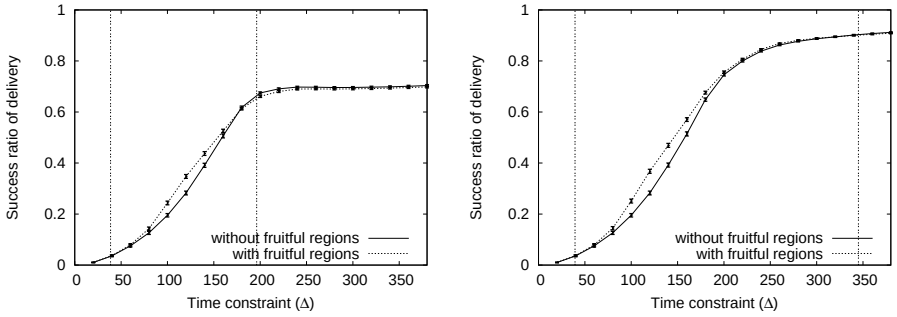
To simulate a dynamic environment with fruitful regions we introduce a particular arrangements of the customers by clusters. First a set of points called the set of cluster centres C is created by randomly selecting points (x, y) in the 2-dimensional space such that these points are uniformly distributed in that space. Then for each cluster centre $(x, y) \in C$ a set of locations $R_{(x,y)}$ is created such that these locations are scattered around the cluster centre by using a Gaussian random distribution with an average distance of τ to choose the diversion from the centre. This way we get clusters with a circular shape. The set of nodes N is defined as $N = \{n | n \in R_{(x,y)} \wedge (x, y) \in C\}$. The set of locations form the nodes of the graph $G = (N, E)$. This graph is a full graph and its edges E are labelled with the costs to traverse them. For each $(n_1, n_2) \in E$, this cost is equal to the Euclidean distance between n_1 and n_2.

A set of loads is randomly generated, which will represent the work that needs to be routed in the time period T. Every load starts at a node and needs to be carried to a central depot, which is located in the centre of the map. Each node is assigned a number of loads, this number is taken from a Gaussian distribution with as average the potential of the cluster and a deviation of loads within a cluster parameter set in Table 1. Clusters are assigned a potential in the same way using the average number of loads per cluster.

Table 1. Parameters of the problem instances, experiment parameters in bold

parameter	*value*
maximum width and height of the map	200×200
number of locations	$\|N\| = 50$
number of clusters	$\|C\| = 5$
spread of locations in a cluster	$\tau = 10$
number of vehicles	$\|V\| = 10$
capacity constraint	$q \in \{1, 2, 3, 4, 5, 6\}$
delivery time constraint	$\Delta \in \{20, 40, \ldots, 380\}$
average number of loads per cluster	5
deviation of the number of loads over clusters	5
deviation of loads per node within a cluster	4
average time spread	$\lambda = 10$ ($\sigma = 50$)

The entry times of loads is chosen by considering all loads one at a time. The i-th load is generated at time $i \times \lambda + N(0, \sigma)$, where λ is the average time spread σ is the deviation of the average time between entry times (see Table 1). This way we spread the loads over time using a normal distribution.

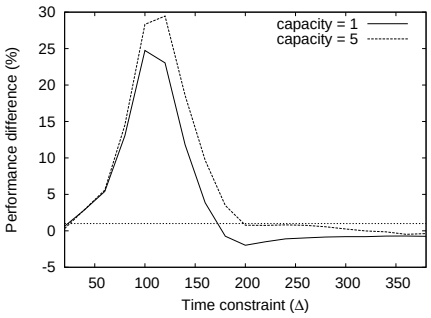


(a) capacity constraint set to one (b) capacity constraint set to five

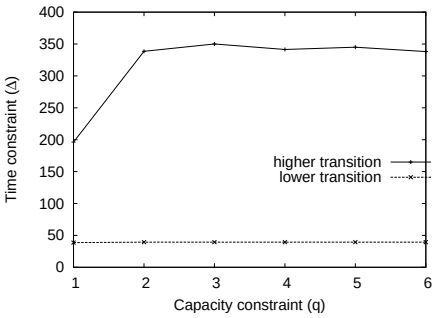
Fig. 2. Average ratio of successfully delivered loads with increasing delivery time for both with and without using anticipated moves. Results for each setting of the time delivery constraint are averaged over 40 problem instances with 10 independent runs of the evolutionary algorithm and include 95% confidence intervals

To show the effect of performing anticipated moves to fruitful regions we fix the capacity q of each vehicle to one and vary the time delivery constraint Δ. By increasing the time available for performing a delivery, we are loosening this constraint, which makes it possible that more loads are transported. We clearly notice three stages, separated by the two vertical lines. Figure 2(a) shows the result for both with and without performing anticipated moves. The first stage, where delivery time is very restricted, and consequently only few loads can be transported. There, anticipated moves will not influence the success ratio. Then, in the next stage, the exploration of fruitful regions shows its potential for increasing the success ratio. Last, when the time restriction is removed further, there is plenty of time to deliver loads, thus the best result is achieved by sticking to routing vehicles for pickup and deliveries only. For the last stage, a vehicle can take a long time to fetch any load, as it will have more than enough time to return to the depot.

With the capacity constraint set to five we get the result shown in Figure 2(b). Most notably, the phase where the exploration of fruitful regions are beneficial widens. In Figure 3(a), we show by how much anticipated moves improves the effectiveness. The largest benefit is found at a time constraint setting of 100 and 120 for $q = 1$ and $q = 5$ respectively, where the success ratio is improved by 25% and 30%. For tight time constraints no significant difference can be noted and for loose time constraints the success ratio is at most 1.8% and 0.8% lower, for the settings of 1 and 5 respectively. This shows that knowledge of the constraint settings is important, but as long as the tightness of the time constraint is not too restrictive, an exploration of fruitful regions is recommendable.



(a) Performance ratios, where the success ratio of with anticipated moves is divided by the success ratio of without anticipated moves



(b) Relating the capacity and time constraints using transition points to determine the boundaries wherein anticipated moves to fruitful regions may improve effectiveness

Fig. 3. Performance ratios and transition points

To take into consideration the effect of both constraints, capacity and time, we show how they relate for the lower and higher transition points. The lower point is where we move from a stage where anticipated moves are not beneficial to a stage where they are beneficial, while the higher point is where this move has the opposite effect. The terms lower and higher correspond to the amount of time available for delivery, where lower refers to small delivery times. For different settings of the capacity constraint (1–6), we determine those transition points, which are shown in Figure 3(b). The results should be interpreted as follows, when the settings of the constraints fall above the line of the lower transition points and below the line of the higher transition points using exploration of fruitful regions by performing anticipated moves is beneficial for the effectiveness of load deliveries.

7 Conclusions

Dynamic constrained optimisation problems play an increasingly more import role in many areas, not only because of faster computers and telecommunication, but because of the increasing demand of customers for flexible and fast service as well. Vehicle routing is a scientifically interesting problem as its static version is known

to be intractable. By moving into the realm of dynamic routing problems, with the given constraint that we should be able to maintain viable solutions, we make this problem even more challenging.

In this paper, we introduced a dynamic routing model that defines a load collection problem, i.e., the pickup and return of loads to one depot. Especially, we introduced a model for taking into account regions with high potentials for the origin of new loads. Furthermore, we have provided an evolutionary algorithm that is able to provide solutions in real-time. Using the evolutionary algorithm, and an extention to let it perform anticipated moves, we have determined the transition points where between the exploration of fruitful regions is of benefit. The potential of using the concept of fruitful regions is evident from the significant increase in the effectiveness for settings within these transition points.

References

1. Bertsimas, D., Simchi-Levi, D.: A new generation of vehicle routing research: Robust algorithms, addressing uncertainty. Operations Research **44** (1996) 286–304
2. Savelsbergh, M., Sol, M.: The general pickup and delivery problem. Transportation Science **29** (1995) 17–29
3. Psaraftis, H.: Dynamic vehicle routing: status and prospects. Annals of Operations Research **61** (1995) 143–164
4. Ghiani, G., Guerriero, F., Laporte, G., Musmanno, R.: Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies. Technical report, Center of Excellence for High Performance Computing, Univ. of Calabria, Italy (2003)
5. de Paepe, W.: Complexity Results and Competitive Analysis for Vehicle Routing Problems. PhD thesis, Research School for Operations Management and Logistics, Technical University of Eindhoven (2002)
6. Laporte, G., Louveaux, F. In: Formulations and bounds for the stochastic capacitated vehicle routing problem with uncertain supplies. Elsevier Science Publishers B.V. (1990) 443–455
7. Laporte, G., Louveaux, F., Mercure, H.: The vehicle routing problem with stochastic travel times. Transportation Science **26** (1992) 161–170
8. Bianchi, L.: Notes on dynamic vehicle routing — the state of the art. Technical report, IDSIA, Galleria 2, 6928 Manno-Lugano, Switzerland (2000)
9. Golden, B., Assad, A.: Vehicle Routing: methods and studies. Elsevier Science Publishers B.V. (1988)
10. Psaraftis, H.: Dynamic vehicle routing problems. [9] chapter 11 223–248
11. Powell, W.: A comparative review of alternative algorithms for the dynamic vehicle routing problem. [9] chapter 12 249–291
12. Powell, W.: A stochastic formulation of the dynamic assignment problem, with an application to truckload motor carriers. Transportation Science **30** (1996) 195–219
13. Gendreau, M., Potvin, J.Y.: Dynamic vehicle routing and dispatching. In Crainic, T., Laporte, G., eds.: Fleet Management and Logistics. Kluwer, Boston (1998) 115–126
14. Solomon, M.: The vehicle routing and scheduling problems with time window constraints. Operations Research **35** (1987) 254–265
15. Branke, J.: Evolutionary Optimization in Dynamic Environments. Volume 3 of Genetic Algorithms and Evolutionary Computation. Kluwer Academic Publishers (2001)
16. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows, part II: Metaheuristics. Transportation Science (to appear)