Online Transportation and Logistics Using Computationally Intelligent Anticipation

Peter A.N. Bosman and Han La Poutré

Centre for Mathematics and Computer Science, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
{Peter.Bosman,Han.La.Poutre}@cwi.nl

Summary. With advances in technology in communication and navigation, the ability to make decisions online (i.e. as time goes by) becomes increasingly important in transportation and logistics. In this chapter, we focus on online decision making in these areas. First, we point out the importance of anticipation when optimizing decision processes online. Anticipation is the possibility to take into account future events and the influence of decisions taken now on those future events. Second, we discuss how computational intelligence (CI) can be used to design approaches that perform anticipation. We illustrate this particular use of CI techniques in two different applications: dynamic vehicle routing (transportation) and inventory management (logistics). In both cases the use of anticipation is found to lead to substantial improvements. This demonstrates our main conclusion that the ability to perform anticipation in online transportation and logistics is very important.

Keywords: Online optimization, Estimation-of-distribution algorithm, Evolutionary algorithm, Vehicle routing, Inventory management.

1 Introduction

Transportation and logistics play an important role in many companies. Optimizing the processes that are involved directly influences a company's efficiency and hence can lead to better revenues. With the advances in technology in communication and navigation, companies can exert an increasing amount of direct control on their transportation and logistics processes. For instance, using global positioning systems the whereabouts of trucks or goods in general can be tracked 24 hours a day. Using global communication, new instructions for moving the goods can be issued at any time. Such technological advances hold a promise to service customers faster because new servicing orders can be given out immediately.

Exploiting these new abilities in the best possible way is therefore an important issue. Traditionally, transportation and logistics problems are optimized using a static model, i.e. plans are made ahead and then executed. If needed, new plans are made for a new period. The main point is that the plans are not made to be adjusted *online*, i.e. while the plans are being executed. Typically,

A. Fink and F. Rothlauf (Eds.): Advances in Computational Intelligence, SCI 144, pp. 185–208, 2008.
springerlink.com

this results in plans that are not very flexible and hence cannot easily accommodate for changes that may be required. This is a myopic, i.e. "near-sighted", approach. The quality of plans is taken only to be how good they are for the current situation. Optimizing plans while keeping in mind that they might need to be adjusted during execution, or optimizing the plans completely online, is however not trivial and is an important field on its own [10, 17].

The optimization problems under consideration are thus dynamic, meaning that they change with time. Moreover, they have to be solved in an online fashion, i.e. as time goes by. Typical examples in transportation and logistics are vehicle routing [21, 27] and inventory management [23, 26]. This type of problem is often hard, even for a single point in time. Also, there is typically not much time between two subsequent decision moments. For these reasons, restarting optimization from scratch is often undesirable. Instead, the tracking of (near-) optima, once they have been found, is often desired. To be able to do this, the optimization algorithm needs to have a proper degree of adaptivity. Evolutionary algorithms (EAs) [16], an important optimization methodology in computational intelligence, are good candidates to this end. The reason for this is that EAs employ a set of solutions rather than just a single solution. Adaptivity is then a virtue of issues such as maintaining diversity around (sub)optima and continuously searching for new regions of interest that may appear over time [10].

Tracking optima alone is not enough however. Tracking optima corresponds to building (optimal) plans only for the current situation and repeating this continuously or whenever something changes. Decisions taken now however have consequences in the future. For instance, a consequence of executing a particular plan may be it does not leave enough flexibility for future requests to be served (e.g. new loads to be picked up). Another example is that refusing service or not being able to service a customer (e.g. sell goods) may lead to decreasing repeated servicing requests in the future. Because of such future consequences of current decisions, a myopic approach can perform poorly in the long run. Anticipation of future situations is needed to be able to make well-informed decisions. Consider the problem of online dynamic vehicle routing. Intuitively, one can construct a more efficient routing, resulting in the delivery of more loads, if one would know beforehand when the loads will become ready for transport. It would then be possible to send a vehicle to a customer that currently has no load available, but it is known that a load will be available at that customer upon arrival of the vehicle. A myopic approach would only consider currently available loads. Alternatively stated, this information allows us to see that a current decision to send a vehicle to pick up a currently available load may in the end not be the best thing to do. However, this optimal information about future introduction of new loads is not readily available. The only option left is to learn to predict it.

In this chapter we discuss how computational intelligence can be used to design approaches that perform anticipation in online optimization. Specifically we combine EAs with another key area of computational intelligence: statistical/machine learning (ML/SL). The ML/SL techniques are used to explicitly

predict for future times the observed values of problem-specific variables and/or the quality of decisions. The EA optimizes decisions not only with respect to the current situation but also future decisions with respect to future, predicted, situations (i.e. it builds a plan). We illustrate this approach for two applications: vehicle routing and inventory management. In both cases the use of anticipation is found to lead to substantial improvements. This demonstrates our main conclusion that the ability to perform anticipation in practical online transportation and logistics is very important.

The remainder of this chapter is organized as follows. In Sect. 2 we sketch the approach of performing anticipation with computational intelligence techniques in some more detail and discuss related literature. We apply the methodology to vehicle routing in Sect. 3 and to inventory management in Sect. 4. We conclude this chapter in Sect. 5.

2 Intelligent Online Optimization by Anticipation

In dynamic optimization, the goal is to optimize some function $\mathfrak{F}$ over a period of time. The variables to optimize over within this time span, represent the decisions to be made (e.g. which truck to use for picking up a certain load). Function $\mathfrak{F}$ can be seen as the *real world*. Function $\mathfrak{F}$ cannot be evaluated beyond the current time t^{now}. The reason why we require anticipation is the existence of time-dependence. Time-dependence means that $\mathfrak{F}$ may depend on previous decisions. If only the current situation is taken into account, the decision that immediately leads to the highest reward is optimal. But because function $\mathfrak{F}$ may change as a result of a decision, it is in general suboptimal to only consider the current situation. Future changes in $\mathfrak{F}$ as a result of decisions taken now must thus be taken into account also. This means that we must optimize $\mathfrak{F}$ not only for the current situation, but also for future situations. Only then does optimization reveal the true value of a certain decision for the current situation. Anticipation is thereby automatically performed. However, this implies that we evaluate $\mathfrak{F}$ also for the decisions pertaining to future situations, which is not possible. The only way in which we can still take into account the future is to *predict* the value of $\mathfrak{F}$. Summarizing, the approach is to build and maintain (i.e. learn online) an approximation of $\mathfrak{F}$ (i.e. $\hat{\mathfrak{F}}$) and to optimize decisions for the present and for the approximated future simultaneously using the approximation. Approximation function $\hat{\mathfrak{F}}$ can be seen as a *simulation* of the real world.

The nice thing about this approach is that under the condition of perfect prediction, optimal decisions can be taken. For a typical application it is not required that the entire function needs to be learned. Instead, only specific parameters need to be estimated. In logistics settings, typical parameters to be learned include the rate of customer demand or parameters describing the distribution of customer demand.

Some parallels can be drawn here with other research found in literature. Optimizing future, simulated situations is not the only approach to performing anticipation. Anticipation can also be performed by heuristically changing $\mathfrak{F}$

into $\tilde{\mathfrak{F}}$ so that $\tilde{\mathfrak{F}}$ measures not only the quality of the current plan, but also takes into account additional information such as the flexibility, robustness and sensitivity of a solution. Although this approach typically does not have the property of being able to reach optimality of the original problem, such heuristic adaptations of $\mathfrak{F}$ can still be very effective. Examples of this approach include scheduling [12], where an emphasis is placed on schedule flexibility by scheduling new jobs as quickly as possible to ensure free machines for new, yet unknown, jobs, and vehicle routing [13], where an emphasis is placed on flexibility by allowing vehicles to wait before moving on to the next location.

The idea and importance of optimizing for future, simulated situations when solving dynamic optimization problems, is in itself not a new idea. In most problems related to real-world applications, the dynamism is caused by stochasticity (e.g. the appearance of new customers is often a Poisson process). In the expectation method [14], multiple future scenarios are sampled. For each possible decision d that can be made for the current situation, an optimal path of future decisions is computed for each of the sampled scenarios that starts with decision d. The decision $d^\star$ for the current situation that leads to the highest expected value of the profit (i.e. average) is then actually taken. The expectation method bears much resemblance to the sample average approximation method for non-dynamic stochastic programming where the quality of a solution consists of a deterministic part a and a stochastic part. The deterministic part gives the (immediate) quality of a plan. The stochastic part gives a penalty for changing the plan to accommodate as best possible the scenario that has actually become reality. To find the expected best solution, one must average over many scenarios. The sample average approximation method can be applied for instance to stochastic vehicle routing [31]. It has been shown recently that the expectation method can provide high-quality solutions in the expected-value sense [22]. It is important though that the expected difference between the optimal choice for any scenario and the optimal expected-value choice does not become too large. It has already been argued that this assumption is satisfied in most real-world problems. It has also been proved to be the case for a real-world problem (packet scheduling) [22]. Because the expectation method requires optimization for many combinations of decisions and scenarios, methods that approximate the expectation method have also been developed [4, 5]. These methods are typically faster, but result in solutions of a lesser quality.

In most of the approaches in the literature, the stochasticity is assumed to be known so that scenarios can be sampled. Hence, there is no learning. Also, the application of optimizing future scenarios so far has been limited to sampling scenarios beforehand and then optimizing the future decisions in these scenarios. Time-dependence is thus explicitly not considered in full. The type of time-dependence that is tackled is the direct influence of one decision on another decision. For example, a decision to drive to customer c_i puts the truck at that location instead of the location where it would otherwise have stayed. What is not taken into account is the possible influence of a decision on the future response of the system, e.g. the stochastics. For example, deciding to drive to

customer c_i may lead to a higher frequency of new orders from customer c_i in the near future. With the existence of time-dependence of this kind, sampling events in a scenario beforehand is unacceptable because the events may change as a result of decisions made.

The general approach requires a minor modification to allow for time-dependence in any of its forms to be tackled. Optimization with scenarios can be done by randomly choosing $N^{\text{scenarios}}$ random seeds and then optimizing the simulation for each random seed. This essentially makes the approximation (i.e. simulation) deterministic, allowing it to be re-evaluated under the exact same circumstances during optimization. The computational implications of this change are however not minor. Finding a solution (i.e. future trajectory) for each and every scenario requires solving an interactive problem (i.e. an online dynamic optimization problem without stochastics). To be able to do this efficiently complex algorithmic design is required. Alternatively, exhaustive search can be used, but this is very inefficient. For this reason the problem instances solved using global (enumerative) optimization methods are relatively small. Also, a problem exists with most methods if the decision to be made concerns *continuous* (e.g. real-valued) variables. It is then not possible to optimize decision trajectories for all possible values for the current decision. It is not even possible to optimize the entire trajectory and then choose for the current situation the decision that has maximum average profit. For continuous decision variables it is namely not likely that optimal values will be the same in different scenarios. Discretization can be a solution, but doing this properly is typically very hard.

Clearly, there is a need for computationally intelligent algorithms that are able to come up with good solutions at any time and are not restricted to solving discrete optimization problems with only a few alternative decisions to choose from at any point in time. Evolutionary algorithms offer a way of doing just that as they are a means of performing optimization by continuously adapting a set of solutions. Thus EAs always have a solution available. Moreover, EAs are known to be able to successfully optimize many different types of solution. In the following sections we shall give two examples of using EAs to solve dynamic optimization problems online in the area of transportation and logistics. In both applications, the EA is continually run to evolve a plan. A plan can be either a list of decisions (yet) to be executed or a strategy on the basis of which decisions are made (i.e. what to do under which circumstances). Whenever a plan is evaluated, it is not only evaluated for the current situation, but also for the predicted future.

3 An Application to Transportation: Vehicle Routing

In this section we focus on the dynamic vehicle routing problem. In this problem, routes have to be planned for a fleet of vehicles to pick up loads at customers. The problem is dynamic because the loads to be transported are announced while the vehicles are already on-route [15].

A few studies currently exist in which information is used about future loads [6, 7, 13, 18, 24, 25, 30]. Most approaches employ a waiting strategy. For each vehicle, upon its arrival at a customer, a waiting window is defined within which a new load is expected to arrive at that customer or at a nearby customer. During that waiting period, the vehicle does not move because it anticipates on having to move only a little in the near future to pick up a new load. In this section, similar to [30], we opt for an approach in which the vehicles keep driving, unless they are at a centrally located depot. The rationale behind this idea is the principled notion that as long as there are loads to be transported, we do not want to have any vehicles waiting around. To move the vehicles as efficiently as possible, we propose to learn the distribution of load announcements at the customers. We use this information to predict the number of future expected loads at a certain customer. By directly integrating this expected value into the fitness of solutions, i.e. vehicle routes, the EA that we use here is able to make informed decisions about anticipated moves (i.e. moves to customers that currently do not have a load ready).

3.1 Problem Definition

The definition of the dynamic vehicle routing problem that we use here is the same as the one used by Van Hemert and La Poutré [30]. Here we shall restrict ourselves to an intuitive, yet concise, description of the problem at hand. Exact mathematical details can be found in [30].

A set of customers is predefined. Each customer has a certain location defined by 2D coordinates. The distance between two locations is the Euclidean distance. The goal in solving the problem is to deliver as many loads as possible. Each load has to be picked up at a certain customer and must be delivered to the central depot. A load has a certain announcement time (i.e. the time from which it is available for pickup). Each load must be delivered to the depot within a certain delivery window, starting from the moment of announcement. Beyond this delivery window the load is no longer accepted by the depot. The size of the delivery window is fixed and is denoted Δ.

To transport the loads, a fleet of vehicles is available. All vehicles have the same capacity. All loads have the same size. Both the size of the loads and the capacity of the vehicles is integer. Initially, all vehicles are at the depot.

At any time t^{now}, the solver must be able to return a list of actions to be performed; one separate action for each vehicle in the fleet. Actions are either to go and pick up a load at a customer, to go to a certain customer without a pickup assignment (i.e. an anticipated move) or to go back to the depot to drop off all loads that are currently being carried.

To ensure that loads are only picked up if they can be delivered on time and to furthermore ensure that loads that have actually been picked up are indeed delivered on time, constraints exist to ensure that such solutions are infeasible. The optimization approach must now only return feasible solutions.

3.2 Optimization Approach

The Dynamic Solver

The dynamic solver updates the optimization problem whenever a change occurs, i.e. when a new load becomes available for pick up. In addition, the dynamic solver controls the EA. It runs the EA and announces changes to the EA so that these changes may be accounted for in the solutions that the EA is working with. It also requests the currently best available solution from the EA whenever changes occur and presents that solution to the real world as the plan to be executed. In our case, the problem changes whenever a new load is announced, whenever a load is picked up or whenever loads are delivered. In addition, the currently executed solution changes whenever a vehicle arrives at a customer, regardless of whether a load is to be picked up there.

The EA is run between changes (also called events). In practice, the time that is available for running equals the time between events. Because computing fitness evaluations takes up most of the time, in our simulated experiments we ensured that the number of evaluations that the EA was allowed to perform between two subsequent events is linearly proportional to the time between events in the simulation. For each time unit of the simulation the EA may perform one generation. Since the population size will be fixed, the EA will thus perform a fixed number of evaluations in one simulation run.

The whole simulation operates by alternatively running the EA and the simulated routing problem. The routing simulator calculates when the next event will occur, e.g., a vehicle will pick up or deliver a load, or, a load is announced for pickup. Then, the EA may run up until this event occurs. This way we simulate an interrupt of the EA 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. Afterward, the individuals of the EA are updated by removing assignments that are no longer applicable (i.e. delivered loads or loads that have passed their delivery window) and by adding assignments to pick up loads that have recently been made available.

Base EA: Routing Currently Available Loads

With the exception of the selection method, the base EA that we use is the same as the one used by Van Hemert and La Poutré [30].

Representation: The representation is a set of action lists, one separate list for each vehicle in the fleet. An action list describes all actions that the vehicle will perform in that order. In the case of the base EA, this action list contains only pickup actions. The first action in the list for a specific vehicle is currently being executed by that vehicle. Properties such as the number of loads that a vehicle already carries is stored in the simulation and is not subject to search.

New loads: Whenever new loads are announced, these loads are injected randomly into the action list of a single vehicle in each member of the population.

Variation: Only mutation is considered. Two vehicles are chosen randomly. These two vehicles may be the same vehicle. From these two vehicles, two actions from their lists are chosen randomly. These actions are swapped. This operator allows visits to customers to be exchanged between vehicles or to be re-ordered in the route of a single vehicle directly.

Selection: The selection scheme that we employ ensures elitism. We use truncation selection to select half of the population. The other half is discarded. Using variation, the discarded half is replaced with new individuals. Hence, elitism of the best half of the population is employed. Although this selection operator is rather strict, enough diversity is introduced as a result of mutation and the random introduction of new loads. As a result, selecting more strictly allows the EA to weed out bad solutions more efficiently.

Decoding: It is important to note that as a result of load introduction and of variation, action lists may come to represent an infeasible solution. For example the action list may cause the vehicle to be on-route too long for the loads that it is carrying to be returned to the depot within the delivery window. For this reason a decoding mechanism is used that decodes the representation into a valid solution, i.e., a solution where none of the constraints are violated. The representation itself is not altered. Assignments that violate one or more time constraints are ignored upon decoding. When a vehicle reaches its capacity or when adding more assignments will violate a time constraint, the decoder inserts a visit to the depot into the action list. Afterward, this vehicle may be deployed to service customers again. This procedure is the same as used in [20]. 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.

Fitness: The fitness of an individual corresponds to the number of loads that is returned to the depot, i.e. the number of loads picked up when executing the current decoded action lists for all vehicles. It should be noted that this representation already provides, in part, a way to oversee future consequences of current decisions. To see this, note that the only decision required to be taken at each point in time from the problem's perspective is what to let each vehicle in the fleet do next. By having a list of actions to perform after the first next move, planning ahead is made possible, which consequently allows for more efficient routing. However, this anticipation at time t^{now} covers only the non-stochastic information about the problem that is available at time t^{now}. It might be possible to improve the benefits of anticipation further by considering some of the stochastic information about the problem at time t^{now}. This is the goal of introducing anticipated moves.

Enhanced EA I: Implicit Anticipated Moves

In this approach, there is no explicit link between making an anticipated move and the expected reward to be gained from that move [30]. Instead, the mechanism behind anticipated moves is implicit and focuses on ensuring that anticipated moves do not cause the resulting routing to violate any constraints. Ultimately, this results in slight deviations from routes that are built based upon currently available loads, otherwise the loads will not be returned to the depot in time. It is only over multiple generations and over time that in retrospect an anticipated move can be found to have been advantageous.

Variation: To guide the introduction of anticipated moves, an anticipated-move-rate α is used. Upon mutation, this parameter represents the probability of introducing an anticipated move into the route of a single vehicle. Similar to mutation in evolution strategies [2], this parameter is subject to self-mutation.

Fitness: To prevent selection of individuals with a large α that cause many constraint-violating anticipated moves to be introduced, the fitness function is extended with a penalty term. The penalty term grows linearly with the number of constraint-violating anticipated moves in a solution.

Enhanced EA II: Explicit Anticipated Moves

The results reported in [30] already indicate an improvement over the base EA for certain values of the delivery window Δ. However, there is no directly apparent reason that the anticipated moves will actually result in the collection of more loads. A bigger improvement is to be expected if a proper, direct and explicit reason for making anticipated moves is introduced. To this end, we opt for an explicit means of anticipation. We still use the basic strategy of introducing anticipated moves randomly, i.e. we use the same variation technique. To bias the search toward feasible anticipated moves with a positive number of expected future loads, we alter the fitness function.

Fitness: First, assume that we have an oracle that can tell us for each customer exactly when future loads will become available at that customer. In that case the fitness of a decoded action list can be computed by not only counting the number of loads that are picked up along the route as a result of premeditated pickup actions, but also the number of loads that are available at customers upon arrival there. Care must be taken that the capacity of the vehicle is not exceeded in computing the number of additional loads. Moreover, only the loads that can still be brought back to the depot on time should be counted. Also, each load should only be counted once to avoid overrating the goodness of anticipated moves when two or more vehicles have planned a visit to the same customer. As we now know exactly how fruitful certain anticipated moves are, a much more efficient search of anticipated moves becomes possible.

In practice we do not have such perfect information. For each customer we therefore propose to estimate the distribution of the time between two subsequent loads

becoming available for transport. To estimate this distribution, we use the normal distribution. How to compute maximum-likelihood estimates for the normal distribution is well known from the literature [1, 29]. The expected number of loads that will become available at a certain customer c_i between the current time t^{now} and the time t^{arrive} of arrival of a vehicle at c_i is just $(t^{\text{arrive}} - t^{\text{now}})/\mu^{c_i}$ where μ^{c_i} is the mean of the distribution of the time between two subsequent loads at customer c_i. Similar to the case of perfect information we must only count the expected loads that can still be brought back to the depot in time. Also the capacity of the vehicle and the possibility of multiple vehicles planning an anticipated trip need to be taken into account. This can be done in the same way as in the case of perfect information.

3.3 Experiments

Experimental Setup

In practice, customers are often clustered into regions as opposed to scattered around uniformly [28]. We therefore use a particular arrangement of the customers by clusters, similar to the arrangement used in [30]. First a set of points called the set of cluster centers 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 center $(x, y) \in C$ a set of locations $R(x, y)$ is created such that these locations are scattered around the cluster center by using a Gaussian random distribution with an average distance of τ to choose the diversion from the center. This way we get clusters with a circular shape. The set of customer 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 weighted with the costs to traverse them. For each $(n_1, n_2) \in E$, this cost is the Euclidean distance between n_1 and n_2.

A set of loads is randomly generated, which represents the work that needs to be routed. Every load starts at a customer node and needs to be carried to a central depot, which is located in the center of the map. Each customer generates loads where the time between two subsequent loads is normally distributed with a mean of μ^{Loads} and a standard deviation of σ^{Loads}. Typically customers are manufacturers. Therefore, the internal process of preparing loads is often quite regular. The larger σ^{Loads}, the less regular the process is assumed to be and the more randomly generated the loads will appear to be.

We have randomly generated 25 problem instances and have run the EA without anticipation, the EA with implicit anticipation, the EA with optimal-information anticipation and the EA with learned-information anticipation for $1 \cdot 10^5$ time units. We have varied the standard deviation of the time between subsequent loads, the delivery window and the capacity of the vehicles. An overview of all parameters used in our experimental setup is given in Table 1.

Table 1. Parameter settings used in our experiments

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	$q \in \{1, 5\}$
Delivery time constraint	$\Delta \in \{20, 40, \ldots, 400\}$
Average time spread of loads	$\mu^{\text{Loads}} = 400$
Standard dev. time spread of loads	$\sigma^{\text{Loads}} \in \{20, 40, \ldots, 200\}$

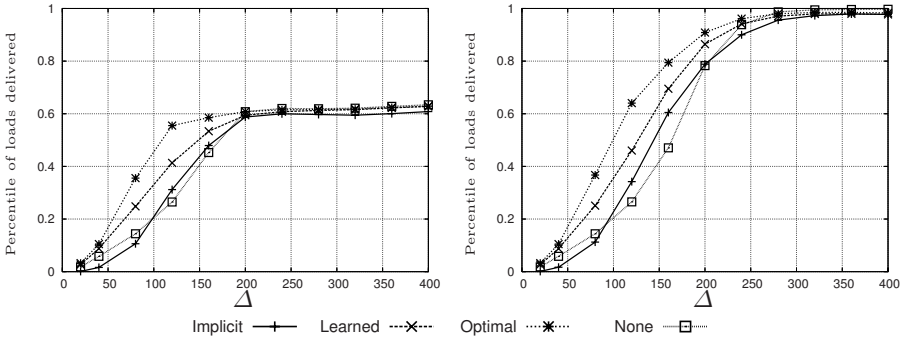

Fig. 1. Routing efficiency in percentage of loads delivered as a function of the delivery window for all EAs and a standard deviation of the time spread of the loads of $\sigma^{\text{Loads}} = 40$. Vehicle capacity is 1 in the left graph and 5 in the right graph.

Results

Figure 1 shows the efficiency of the various EAs with respect to the problems in our test suite for a standard deviation of the time spread of the loads of 40. There is a clear shift in problem difficulty when varying the length of the delivery time window. If this time window is very small, anticipatory routing only pays off if one is certain that there will be loads that can be picked up upon arrival at a certain customer. The number of loads that can be picked up and delivered on time is so small that uninformed anticipatory moves directly cause a drop in the number of loads that could have been delivered otherwise. Indeed, if the learned information or the perfect information is used, an improvement can be found over not using anticipatory moves, where the perfect information of course leads to the biggest improvements. For an average Δ there is much room for improvement. Indeed all anticipatory approaches are capable of obtaining better results than the non-anticipatory approach. However, the use of explicit learning and predicting the announcement of new loads is able to obtain far

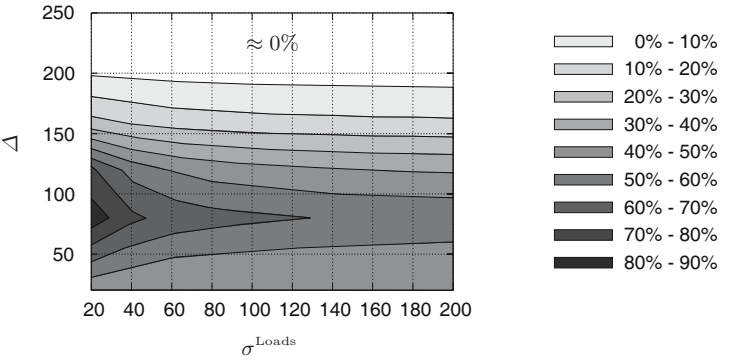

Fig. 2. Relative performance increase of our proposed anticipation-by-explicit-learning EA over the non-anticipatory EA for various values of the delivery window and the standard deviation of the time spread of the loads

better results than when the implicit means of anticipation is used. If the delivery window becomes too large, there is ample time to fully use the capacity of the fleet to the maximum and hence there is no longer any difference between using anticipation and using no anticipation. The problem thus becomes easier.

Figure 2 shows a contour graph of the relative performance increase that can be obtained when using our explicit prediction approach to anticipation as compared to the EA that does not use anticipatory moves. This height-map shows clearly the phase-shift with respect to the delivery time window. There are clear bounds within which an improvement can be obtained. This graph also shows the influence of the randomness of the problem. The larger the standard deviation of the time between subsequent loads, the smaller the performance increase becomes. Clearly, the largest performance increase can be obtained if the variance goes down to 0, which corresponds to the case of using the optimal information. Although the best results only comprise a small area of the graph and thus correspond only to a specific type of problem settings, the range of problem settings for which an improvement can be obtained is large and rather robust with respect to an increase in randomness. Hence we can conclude that our explicit anticipatory approach provides a robust means of improving the quality of online dynamic vehicle routing that is generally speaking preferable compared to an implicit means of anticipatory routing.

4 An Application to Logistics: Inventory Management

General Description

We employ a commonly used definition of inventory management (IM) problems [26]. In Fig. 3 a schematic overview is given. Buyers, also called customers and denoted C_i, buy goods from a vendor. The number of buyers is denoted n_c. The number of goods and the frequency of buying are called the demand and is

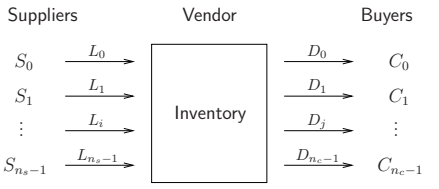

Fig. 3. Schematic overview of inventory management

denoted D_i for customer C_i. To prevent going out of stock, the store keeps an inventory. Inventory must be replenished from time to time. Because delivery of new stock from the store's suppliers also takes time (called the lead time, denoted L_j for supplier S_j), the replenishment order must be placed before going out of stock. The number of suppliers is denoted n_s.

Base Definition

Although IM poses problems that are continuous through time, when solving these problems we typically discretize them. In this chapter, we will use a discretization of time into units of a minute. Let $M(t)$ be the money that the vendor has at time t. The decisions to be taken are how much to order and which suppliers to order from at a any given point in time. Hence, a general formulation of IM is

$$\max_{\boldsymbol{x}(t)} M(0) + \left\{ \sum_{t=0}^{t^{\text{end}}} \Delta M(t) \right\} \tag{1}$$

where $\Delta M(t)$ is the change in the money of the vendor at time t, $M(t)$ depends on the decisions $\boldsymbol{x}(t)$ and t^{end} is the end of the planning horizon. The expenses of the vendor are the costs of holding the inventory and the ordering of new supplies. The income of the vendor is based on sales made:

$$\Delta M(t) = \mathit{Sales}(t) - \mathit{HoldingCost}(t) - \mathit{OrderingCost}(t) \tag{2}$$

The holding cost can be computed per time unit and depends on the size of the inventory $I(t)$. Typically, holding costs are a fixed price p^H per unit of inventory per time unit:

$$\mathit{HoldingCost}(t) = p^H I(t) \tag{3}$$

Typically, the cost of an order that is placed at some supplier is paid for when the order is delivered. The ordering cost at time t therefore depends on earlier decisions $\boldsymbol{x}(t')$, $t' < t$. The cost of a replenishment order at supplier S_i typically consists of two parts: a fixed part c_i^O and a part that increases with the size of the order. Typically, a fixed price p_i^O per unit of ordered goods is charged. Let $x_i^{\text{OrderQuantity}}(t)$ be the quantity of goods ordered from supplier i at time t and let

$L_i(x_i^{\text{OrderQuantity}}(t'))$ be the time it takes supplier S_i to deliver that order, then we have:

$$OrderingCost(t) = \sum_{t'<t} \sum_{i=0}^{n_s-1} o(i,t',t) \tag{4}$$

where

$$o(i,t',t) = \begin{cases} c_i^O + p_i^O x_i^{\text{OrderQuantity}}(t') & \text{if } x_i^{\text{OrderQuantity}}(t') > 0 \\ & \text{and } t' + L_i(x_i^{\text{OrderQuantity}}(t')) = t \\ 0 \text{ otherwise} & \end{cases}$$

Note that in an implementation, it is not efficient to use Equation 4 directly. Instead, an event queue can be used to check whether a new supplier order has arrived at time t and hence whether payment is due.

The income of sales at time t depends on whether the demand of a buyer at time t can be met. Only if there are enough goods in inventory, a sale is made. Partial sales are typically excluded [26]. The goods are sold at a fixed unit price p^S. Let $D(t)$ be the demand of the buyers at time t, then we have:

$$Sales(t) = \begin{cases} p^S D(t) \text{ if } D(t) \leq I(t) \\ 0 \qquad \text{otherwise} \end{cases} \tag{5}$$

We note that there are many other ways in which the above costs and gains can be computed, but most are just minor variations of the equations presented above. Also, other costs and gains besides the ones mentioned above may be taken into account such as the directly computable cost of having more demand than inventory (called excess demand cost) and the subsequent possible loss of sales (called lost sales cost). Such functions are similar to the ones above. Although they indeed make the model more involved, it does not increase the level of problem difficulty, especially for a general problem solving approach that we shall use for this type of problem (see Sect. 4.1).

Although now $\Delta M(t)$ has been defined, there are still functions in the above definitions that are left undefined, namely $I(t)$, L_i and $D(t)$. The inventory (or stock) level $I(t)$ usually is assumed to have the property $I(0) = 0$, but this is not a requirement. In addition, the inventory level rises with $x_i^{\text{OrderQuantity}}(t')$ at time $t' + L_i(x_i^{\text{OrderQuantity}}(t'))$ (i.e. when the supplier order arrives). The inventory level falls with $D(t)$ when a sale is made at time t. The formulations for the rise and fall of $I(t)$ are very similar to those of $\Delta M(t)$. Function L_i for the delivery time for an order placed at supplier i typically follows a certain probability distribution. The same holds for the demand function $D(t)$. The demand function has two sources of stochasticity however: both the quantity Dq of the demand and the frequency of the demand Df. In this chapter we shall assume all these functions to be normally distributed at any time t with means of respectively $\mu^L(t)$, $\mu^{Dq}(t)$ and $\mu^{Df}(t)$ and variances of respectively $\sigma^{2,L}(t)$, $\sigma^{2,Dq}(t)$ and $\sigma^{2,Df}(t)$ where $\mu^{Df}(t)$ and $\sigma^{2,Df}(t)$ describe the distribution of the time between two subsequent customer visits. Note that all these distributions are restricted to $\mathbb{R}^+$.

Time-dependence plays an important role even in the base definition of IM. The decision of whether or not to place a replenishment order at a certain point in time has great future consequences because it determines future inventory levels. Also, a decision to place an order at a supplier leads to a future event (delivery of goods) that is a response to placing the order. Still, although time-dependence is already important here, it is of a rather trivial nature because the only function that is affected by decisions made earlier is the level of the inventory. Given a fixed demand, a single supplier, and a fixed lead time for that supplier, the best strategy for placing replenishment orders is straightforward to compute [26]. Although this is already no longer possible if there is more than one supplier [23], an extension of the model that defines a second, but also very practically relevant, level of time-dependence, makes the problem even harder.

Extended Definition: Customer Satisfaction

Customer satisfaction is important in doing business with customers. A higher level of customer satisfaction will most likely result in a growing frequency of customer transactions, either from the same customer or new customers as satisfied customers will spread the word. In our model this means that whether or not a customer is satisfied when requesting goods from the vendor influences the stochastic model that underlies the customer demand behavior. We can integrate this in the above model by changing the parameters that describe the distribution of the time between two subsequent customer visits, $\mu^{Df}(t)$ and $\sigma^{2,Df}(t)$. If a sale can be made (there is enough inventory, see Equation 5), the customer frequency increases and thus the time between two subsequent customer visits decreases. If the customer cannot be satisfied (no sale is made), the frequency decreases:

$$\begin{aligned} \mu^{Df}(t+1) &= \max\left\{1, C(t)\mu^{Df}(t)\right\} \\ \sigma^{2,Df}(t+1) &= \max\left\{1, C(t)\sigma^{2,Df}(t)\right\} \end{aligned} \tag{6}$$

$$C(t) = \begin{cases} \frac{1}{2} \text{ if } D(t) \leq I(t) \text{ and } D(t) > 0 \\ 2 \text{ if } D(t) > I(t) \text{ and } D(t) > 0 \\ 1 \text{ otherwise } (D(t) = 0) \end{cases} \tag{7}$$

The influence of customer satisfaction now is one of true time-dependence, which can be seen as follows. A decision whether or not to order new goods from a supplier has a future effect on the size of the inventory. Depending on the level of the inventory, a sale can either be in the future made or not. Although this is already a form of time-dependence, customer satisfaction brings a secondary level of time-dependence. Whether or not a future sale can be made influences further future sales indirectly because of an altered customer frequency. This secondary effect is very important however, because it determines the rate of change in future profits. If a higher frequency of customer visits can be met as a result of proper inventory management, profits can be made faster. However, this

type of time-dependent influence is often overlooked in literature [3]. Without overseeing the future effects on the frequency of customer visits, the strategy optimizer will see no need to change the strategy beyond one that meets with the current expected rate of customer frequency. Although this does not mean that profits become losses, the strategy can clearly not be optimal.

4.1 EA Design

Many models exist for simple to hard problems in IM. For the simplest problems, exact optimal strategies are known. For practical problems however, there are typically multiple store suppliers to choose from with different delivery times, order magnitudes and costs. Also demands and lead times tend to be stochastic rather than deterministic. Although these aspects increase the benefit of a non-myopic view, they also make the problem harder to solve [23]. Consequently, no exact optimal strategies exist for the harder and more general cases. For specific cases, specific heuristics exist. There is no general flexible approach however that is applicable to a variety of IM problems. Here we again follow the principled idea of optimizing the future to oversee the consequences of current decisions.

We again use EAs as the base optimization technique. We take a slightly different approach to finding the decisions than we did in the previous section for vehicle routing however. It is common practice in IM to find a strategy instead of individual decisions. The strategy that we employ is a common one in IM. The strategy is a so-called (s, Q) strategy [26]; s is called the re-order point and Q the order-up-to size. One such strategy is used for each supplier. Hence, the genotype contains $2n_s$ real values to be optimized, where n_s is the number of suppliers. If the stock drops below the re-order point s_i of supplier i, and no order is currently outstanding for supplier i, a new order is placed at supplier i of size $Q - \mathit{stocklevel}$. Thus, in the case of two suppliers, if an order from the cheaper supplier is running late, the stock level will drop further and the rule for the more expensive, emergency supplier becomes active. It is not known whether this strategy can be optimal for the case of two suppliers, but it is an often used, sensible choice.

The EA thus finds a strategy. In addition to a population, a current best strategy is maintained. This allows the EA to be run continuously. Whenever a decision needs to be made, the current best strategy can be applied. Evaluation of a strategy is done by running that strategy in the simulation multiple times using different random seeds (i.e. in different scenarios). Note that in the previous section, we used the expected values of the distributions for prediction instead of averaging over multiple scenarios. It was recently shown that the optimization of strategies using EAs works better when averaging over multiple sampled future scenarios is used instead of the expected value [9]. The quality of a strategy is measured by its average evaluation value, averaged over all sampled future scenarios. The variance is also stored. The variance is required to compare the best strategy in the population with the current best strategy. Because multiple random seeds are used, corresponding to multiple drawings from the probability distribution that underlies the problem, statistical hypothesis tests are required

to be certain that an improvement has been obtained. The statistical hypothesis test that we used in our experiments is the Aspin-Welch-Satterthwaite (AWS) T-test at a significance level of $\alpha = 0.05$. The AWS T-test is a statistical hypothesis test for the equality of means in which the equality of variances is not assumed [19].

The EA must optimize the parameters of the (s, Q) strategies. Since these are real values, we use a real-valued EA. Specifically, we use a recent Estimation-of-Distribution Algorithm (EDA) called SDR-AVS-IDEA [8]. EDAs estimate the distribution of the selected solutions and sample new solutions from this estimated distribution in an attempt to perform variation more effectively. In SDR-AVS-IDEA normal distributions are used. For this problem we did not learn any covariances. This means that SDR-AVS-IDEA computes for each parameter the mean and variance in the selected set of solutions and resamples new solutions with these parameters using a separate one-dimensional normal distribution for each parameter to be optimized. It should be noted that other EAs can be used as well. Particularly of interest are EAs that have been designed to tackle dynamically changing environments without anticipation such as the self-organizing scouts [11]. The focus of this chapter however is on the extended design of EAs to facilitate anticipation. For the problems at hand, the relatively simple EA that we employ suffices.

4.2 Experiments

Problems

We use four IM problems. For each problem, inventory is to be managed for 129600 minutes, i.e. 90 days. Orders can be placed any minute of the day.

Problem I represents problems of the type for which an optimal strategy can be computed beforehand. There is one supplier and one product. Product quantities are integer. The product is sold to the buyers at a price of 50 and bought from the supplier at a price of 20. A fixed setup cost for each order placed at a supplier is charged at a price of 50. Inventory holding costs are 1 per day per unit. The lead time of the supplier is fixed to 3 days. The demand is fixed to an order of 1 item every hour.

Problem II represents problems for which there is not a known optimal strategy. There are two suppliers. One supplier is cheaper than the other. The more expensive supplier can supply immediately, but costs twice as much. This type of setting is popular in IM research. It is typically known as IM with emergency replenishments and is known to be a hard problem [23]. The second supplier is used only if the stock has become really low and stock outs are imminent. To add to the difficulty of the problem, we have made the lead time of the cheapest supplier both stochastic and periodically changing. The lead time of the slower supplier is normally distributed with mean (in minutes) of $4320\left(\cos((2\pi t)/43200)+1\right)/2$, i.e. it varies between 0 and 3 days and the period-length of the cosine is 30 days. The variance is $1440^2\left(\cos((2\pi t)/43200)+1\right)/2$, i.e. it varies between 0 and 1440

days, corresponding to a maximum standard deviation of 38 days with the same period-length as the mean. The periodically changing lead time causes the optimal strategy to change with time as well. The maximum size of the standard deviation is not very likely to occur in practical situations. The experiments in this chapter are however meant to illustrate the working principles of the framework. The current setup will show whether changes can be anticipated. The demand is now also stochastic. The time between two subsequent orders is normally distributed with a mean of one hour and a variance of 60 hours. The amount ordered is also normally distributed, with a mean of 3 products and a variance of 9 products. For this setting, there are not any known heuristics.

Problem III equals Problem I but has customer satisfaction (Equation 6).

Problem IV equals Problem II but has customer satisfaction (Equation 6).

Algorithmic Setup

We used two different EA settings, a "small" setting and a "big" setting. The small setting corresponds to a situation in which there is only very little time to do optimization and thus the EA resources are small. The big setting corresponds to a situation in which there is more time and thus the EA resources are larger. In the small settings, the population size is 50, scenario-evaluation simulates 10 days into the future, 5 generations of the EA can be done per day, and 10 scenarios are used. In the big settings, all settings are three times bigger. The population size is 150, scenario-evaluation simulates 30 days into the future, 5 generations of the EA can be done every eight hours and 30 scenarios are used. To facilitate the simulation of the future, the EA learns the distribution behind the stochasticity of the buyer using maximum likelihood estimates. The stochasticity of the supplier is assumed to be known. All results were averaged over 100 independent runs.

Results

Problem I: In Fig. 4 the average profit obtained is shown for both EA settings. The approach can be seen to be a scalable technique (also on Problem II) in the sense that allowing more resources results in better solutions. Investing in computing power thus results in a better policy for a vendor. Moreover, even the small settings for the EA lead to very good profits. The maximum profit that can be obtained on problem I is 58888. This profit corresponds to a setting of the strategy ($(s, Q) = (143, 143)$) that is far outside the range in which we initialized the EA ($(s, Q) \in [0, 25] \times [0, 50]$). Out of all settings in the initialization range, the maximum profit is only 25705. The EA is thus also capable of finding much better solutions when initialization is suboptimal. The big EA settings lead to near-optimal results.

Figure 4 also shows the strategies obtained with the big EA settings for both problems in a typical run of the EA. The lack of stochasticity in problem I

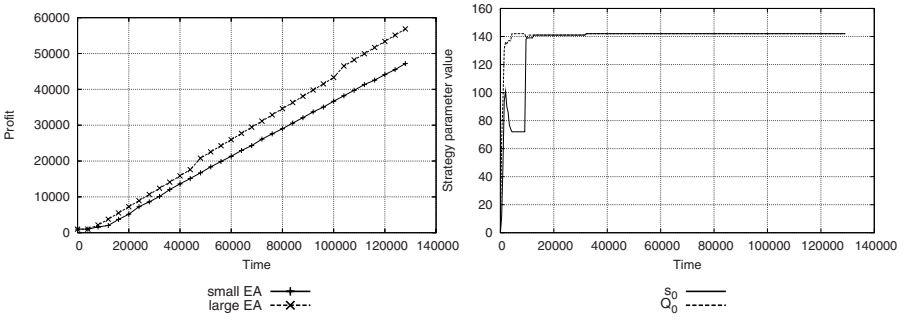

Fig. 4. *Left:* Results on inventory management problem I; *Right:* Strategies evolved in a typical run of the EA on problem I

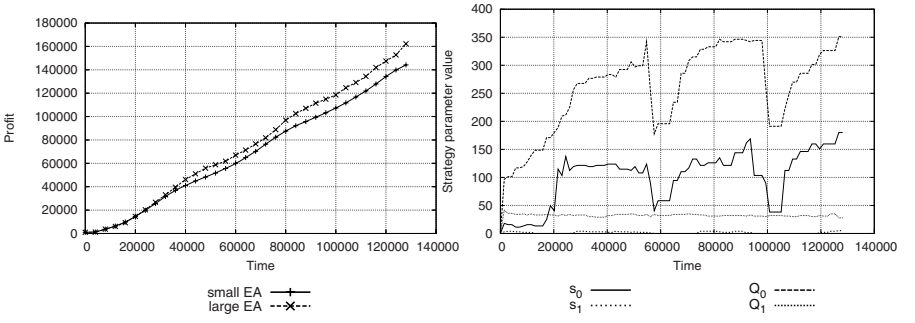

Fig. 5. *Left:* Results on inventory-management problem II; *Right:* Strategies evolved in a typical run of the EA on problem II

translates into finding a stable strategy by the EA very quickly and maintaining that strategy throughout the run.

Problem II: In Fig. 5 the average profit obtained is shown for both EA settings. The profits on problem II are higher than on problem I. The average demand in problem II is three times higher than in problem I. Indeed, the EA is able to obtain a profit of about 3 times higher than on problem I even though problem II is far more difficult.

Figure 5 shows that the adaptive capacity of the EA allows the algorithm to continuously adapt the strategies and find better solutions for the situation at hand as the lead time of the cheapest supplier changes with time. The periodic change of the lead time of the cheapest supplier (S_0) is clearly translated into a periodic change in strategy. When the average and variance of the lead time of the cheapest supplier are small, less products need to be ordered and the threshold can be lower. The threshold for emergency replenishments can even become 0. When the lead time is the largest, emergency replenishments may become

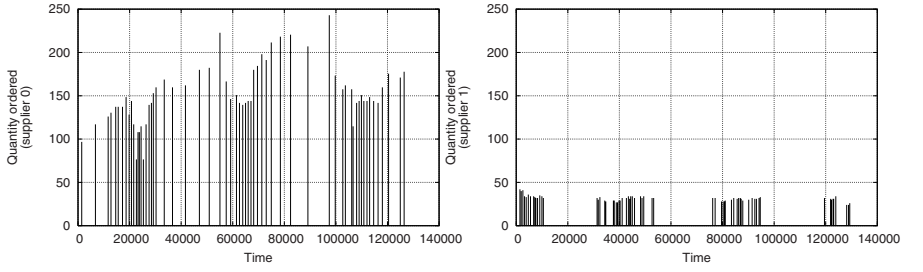

Fig. 6. Quantities ordered in a typical run of the EA on problem II

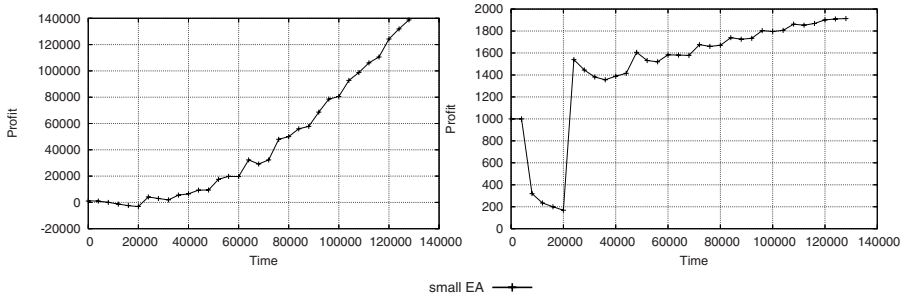

Fig. 7. Results on inventory management problem III with (*Left*) and without (*Right*) anticipation of customer satisfaction

necessary and concordantly, the EA proposes a strategy in which emergency replenishments are made before the stock runs out completely. Also, in this case the re-order point for the cheapest supplier is much higher as is the number of products ordered from that supplier. It can be seen in Fig. 6 that emergency replenishments are indeed made during the periods when the cheapest supplier is less reliable. Furthermore, note that the strategies are not exactly the same in each period. Note that while the EA is optimizing strategies, it is also still learning distributions. Learning converges to the true distribution over time. Finally, the periodic change in the lead time of the cheapest supplier can also be seen in the obtained profits in Fig. 5. When the lead time of the cheapest supplier is the smallest, the EA finds a strategy that uses this supplier more and therefore obtains more profit, resulting in a steeper slope of the profit-versus-time graph at these moments.

Problem III: From the results on problems I and II it is now clear that giving more resources to the EA improves the results. For this reason we now only continue our experiments with the small EA settings. Figure 7 shows the average profit that was obtained if the EA is run with anticipation and without anticipation. The difference between the results is very large. Without anticipation,

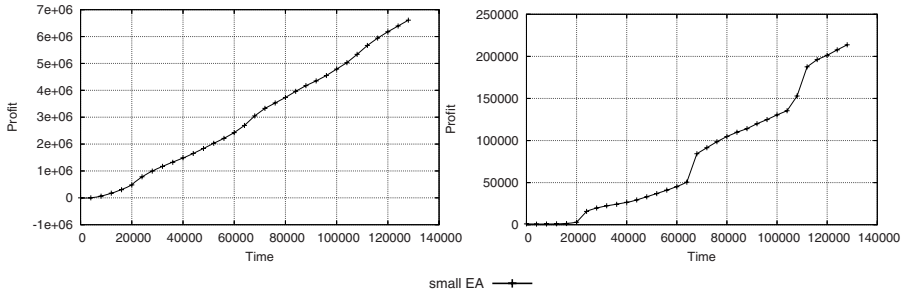

Fig. 8. Results on inventory management problem IV with (*Left*) and without (*Right*) anticipation of customer satisfaction

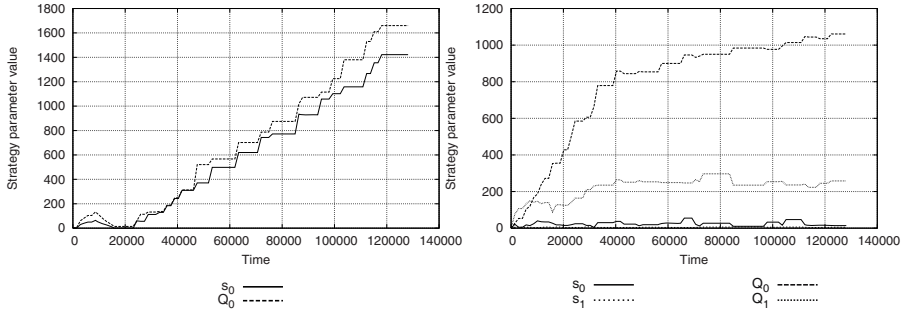

Fig. 9. Strategies evolved in a typical run of the EA with anticipation on problem III (*Left*) and problem IV (*Right*)

the EA is still able to obtain a profit. With anticipation however, the EA notices that having a larger inventory eventually leads to a higher frequency of sales and the result is a much faster growing profit. The growth is exponential until the customer frequency (one customer per minute) can no longer increase.

Inventory growth as a result of ordering more supplies (i.e. a change in strategy) can clearly be seen in Fig. 9. The strategy moves far away from its initialization range and continues to raise the number of goods to order as time goes by until customer satisfaction can no longer be increased (which happens approximately at the end of the run).

Problem IV: A difference that is similar in magnitude can be seen in Fig. 8 between the EA that uses anticipation and the EA that does not use anticipation on problem IV. The main difference with the results on problem III (besides the effect due to the the periodic change in the supply time of the main supplier) is that the increase in the rate of profit is almost immediate. On problem III it

takes some time before the strategy matches the maximum customer satisfaction level. The reason for this difference is the emergency supplier. The emergency supplier can supply goods immediately. Ordering more from the emergency supplier therefore has a much faster effect on the customer satisfaction level. In Fig. 9 it can indeed be seen that the strategy for ordering from the emergency supplier is changed very quickly: an increase in the number of goods to be ordered can be observed. This increase allows to immediately meet with the quickly growing demand of the customers. Meanwhile, the strategy for ordering from the main supplier also increases the number of goods to order, making more profit in the end by meeting the maximum level of customer satisfaction mostly through the supplies bought from the main (and cheaper) supplier rather than the emergency supplier.

5 Discussion and Conclusions

In this chapter we have focused on designing intelligent algorithms for solving dynamic optimization problems in the transportation and logistics domain in an online fashion. We have indicated that it is not enough to only track optima as they shift with time. The optimization algorithm is also required to perform anticipation, i.e. take into account consequences of decisions taken earlier. To this end, we investigated a principled approach in which optimization is performed not only for the current decision but also, simultaneously, for future decisions in future, predicted situations.

We have applied this approach to key problems in transportation and logistics, specifically vehicle routing and inventory management, and found significant improvements over traditional approaches. By analyzing carefully what it is that should be predicted, learning this information from past experience and explicitly incorporating it in anticipating the consequences of current decisions, better results can be obtained than when using no prediction or when using an implicit means of prediction.

In addition to the positive results on the problems in this chapter, one of the most beneficial aspects of the approach that we used is that when the problems are changed the overall approach remains the same and is able to obtain positive results. No in-depth redesigning needs to be done when something changes in the definition of the problem. Often, well-designed heuristics need to be completely redesigned when transferring a problem from theory into practice because of discrepancies between the theoretical case and the practical case and the fact that the heuristics are very problem-specific.

Although there are many important questions still to be answered about anticipation in general such as whether the length of the time-interval that is required to use future predictions over can also be detected online, we can conclude that online dynamic optimization and the use of anticipation represent an important avenue of research.

References

1. Anderson, T.W.: An Introduction to Multivariate Statistical Analysis. Wiley, New York (1958)
2. Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, Oxford (1996)
3. Beamon, B.: Supply chain design and analysis: Models and methods. International Journal of Production Economics 55, 281–294 (1998)
4. Bent, R., Van Hentenryck, P.: Online stochastic and robust optimization. In: Maher, M.J. (ed.) ASIAN 2004. LNCS, vol. 3321, pp. 286–300. Springer, Heidelberg (2004)
5. Bent, R., Van Hentenryck, P.: Regrets only! Online stochastic optimization under time constraints. In: McGuinness, D.L., Ferguson, G. (eds.) Proceedings of the National Conference on Artificial Intelligence – AAAI 2004, pp. 501–506. AAAI Press, Menlo Park (2004)
6. Bent, R., Van Hentenryck, P.: Scenario-based planning for partially dynamic vehicle routing with stochastic customers. Operations Research 52(6), 977–987 (2004)
7. Bent, R., Van Hentenryck, P.: Waiting and relocation strategies in online stochastic vehicle routing. In: Veloso, M.M. (ed.) Proceedings of the International Joint Conference on Artificial Intelligence – IJCAI 2007, Hyderabad, pp. 1816–1821 (2007)
8. Bosman, P.A.N., Grahl, J., Rothlauf, F.: SDR: A better trigger for adaptive variance scaling in normal EDAs. In: Thierens, D., et al. (eds.) Proceedings of the Genetic and Evolutionary Computation Conference – GECCO 2007. ACM Press, New York (2007)
9. Bosman, P.A.N., La Poutré, H.: Learning and anticipation in online dynamic optimization with evolutionary algorithms: The stochastic case. In: Thierens, D., et al. (eds.) Proceedings of the Genetic and Evolutionary Computation Conference – GECCO 2007, pp. 1165–1172. ACM Press, New York (2007)
10. Branke, J.: Evolutionary Optimization in Dynamic Environments. Kluwer, Norwell (2001)
11. Branke, J., Kaußler, T., Schmidt, C., Schmeck, H.: A multi–population approach to dynamic optimization problems. In: Parmee, I.C. (ed.) Adaptive Computing in Design and Manufacture – ACDM 2000, pp. 299–308. Springer, Berlin (1999)
12. Branke, J., Mattfeld, D.: Anticipation and flexibility in dynamic scheduling. International Journal of Production Research 43(15), 3103–3129 (2005)
13. Branke, J., Middendorf, M., Noeth, G., Dessouky, M.: Waiting strategies for dynamic vehicle routing. Transportation Science 39(3), 298–312 (2005)
14. Chang, H., Givan, R., Chong, E.: Online scheduling via sampling. In: Chien, S., et al. (eds.) Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems – AIPS 2000, pp. 62–71. AAAI Press, Menlo Park (2000)
15. Ghiani, G., Guerriero, F., Laporte, G., Musmanno, R.: Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies. European Journal of Operational Research 151(1), 1–11 (2004)
16. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learing. Addison Wesley, Reading (1989)
17. Grötschel, M., Krumke, S.O., Rambau, J. (eds.): Online optimization of large scale systems. Springer, Berlin (2001)
18. Ichoua, S., Gendreau, M., Potvin, J.-Y.: Exploiting knowledge about future demands for real-time vehicle dispatching. Transportation Science 40, 211–225 (2006)

19. Kendall, M.G., Stuart, A.: The Advanced Theory Of Statistics, Inference And Relationship, vol. 2. Griffin, London (1967)
20. Laporte, G., Louveaux, F., Mercure, H.: The vehicle routing problem with stochastic travel times. Transportation Science 26, 161–170 (1992)
21. Larsen, A.: The Dynamic Vehicle Routing Problem. PhD thesis, Technical University of Denmark, Denmark (2000)
22. Mercier, L., Van Hentenryck, P.: Performance analysis of online anticipatory algorithms for large multistage stochastic programs. In: Veloso, M.M. (ed.) Proceedings of the International Joint Conference on Artificial Intelligence – IJCAI 2007, Hyderabad, pp. 1979–1984 (2007)
23. Minner, S.: Multiple-supplier inventory models in supply chain management: A review. International Journal of Production Economics 81(82), 265–279 (2003)
24. Mitrovic-Minic, S., Krishnamurti, R., Laporte, G.: Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows. Transportation Science B 38, 669–685 (2004)
25. Mitrovic-Minic, S., Laporte, G.: Waiting strategies for the dynamic pickup and delivery problem with time windows. Transportation Science B 38, 635–655 (2004)
26. Nahmias, S.: Production and Operations Analysis. Irwin, Homewood (1997)
27. Savelsbergh, M.: DRIVE: Dynamic routing of independent vehicles. Operations Research 46(4), 474–490 (1998)
28. Solomon, M.: The vehicle routing problem and scheduling problems with time window constraints. Operations Research 35, 254–265 (1987)
29. Tatsuoka, M.M.: Multivariate Analysis: Techniques for Educational and Psychological Research. Wiley, New York (1971)
30. van Hemert, J.I., La Poutré, J.A.: Dynamic routing problems with fruitful regions: Models and evolutionary computation. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 690–699. Springer, Heidelberg (2004)
31. Verweij, B., Ahmed, S., Kleywegt, A.J., Nemhauser, G., Shapiro, A.: The sample average approximation method applied to stochastic routing problems: A computational study. Computational Optimization and Applications 24(2-3), 289–333 (2003)