Optimization of large scale parcel distribution systems by the Breeder Genetic Algorithm (BGA)

U. Bartling[*] and **H. Mühlenbein** [†]
GMD - German National Research Center for Information Technology
D-53754 St. Augustin
Germany

Abstract

The routing and scheduling of vehicles and their crews is an area of increasing importance. In this paper, we describe a large scale vehicle scheduling and routing problem which is of great importance to parcel distribution companies. We devise a Breeder Genetic Algorithm capable of dealing with up to 10,000 transportation requests to be serviced by an inhomogeneous fleet of vehicles within a 24 hour time interval. A transportation request is defined as the task to move a loaded container from one depot to another. Since the depots do not send out the same number of containers as they receive, the amount of empty containers available at the depots has to be balanced. The optimization task, therefore, is twofold: determine appropriate balance tours and find a low cost schedule for the fleet of vehicles.

1 INTRODUCTION

The routing and scheduling of vehicles and their crews is an area of increasing importance. Often, the problems described in the scientific literature oversimplify the ones that occur in practice. In this paper, we describe a large scale practical vehicle routing problem which is of great importance to parcel distribution companies.

The generic problem, at least for Germany and Europe, can be described as follows. The parcels are picked-up from customers or post offices by small vans, and delivered to a nearby depot. Upon arrival, the parcels are sorted into containers according to their destination depots. During the night these containers are transported between the depots. The containers are very large, normally a truck can transport up to two containers. Early in the morning packets are delivered to the customers (or post offices).

Sometimes the number of parcels to be transported between two depots is so small that a container would be almost empty. In that cases all parcels from a depot to destinations with low traffic are consolidated and sent to a hub. At the hub the packets are sorted, packed into containers again and delivered to their destination depot.

If the traffic is restricted to point to point traffic, then the optimization problem is straightforward. But in practice round tours, where a truck visits two or three depots, reduce the transportation cost substantially. If possible two trucks exchange their containers at a meeting point half along the way between two depots.

An important constraint is the utilization of the depots. If containers are delivered too late, they cannot be sorted in time. Furthermore, the traffic is not symmetric. A good vehicle routing system has to take care of empty containers.

This is a generic description of the problem. In Germany there are at least three major companies for parcel distribution, the largest one being Deutsche Post AG. To give a rough idea of the size of the optimization problem: Deutsche Post AG runs about 46 depots (some of them belong to assorted customers like mail order services, who only send parcels, but do not receive any). On the average about 2600 containers have to be transported each night. During Christmas time the traffic is substantially larger.

To the best of our knowledge we are not aware of any work addressing this problem in its entirety.

The outline of the paper is as follows. In sections 2 and 3 the optimization problem is specified in detail. Then the genetic representation and important

[*]ulrich.bartling@gmd.de
[†]heinz.muehlenbein@gmd.de

genetic operators are explained. Hill-climbing strategies are described in section 5. Results using real life data are presented in the final section. Since the majority of the data is confidential we can report only a small portion of the results.

2 THE PROBLEM

The distribution of goods is often organized as a three step process:

1. Collect the goods in the service area of a depot.
2. Pack the goods into containers and transport them to their destination depot.
3. Distribute the goods in the service area of the destination depot.

Steps 1 and 3 represent the classical Single Depot Vehicle Routing Problem (SDVRP). In step 2 we are faced with a transportation problem belonging to the class of task or trip routing. For a classification of VRP see [Bodin 1981], [Desrochers 1990]. Each container to be transported, i.e. each trip, is characterized by a place of origin, a destination, and a time window during which the transportation must be carried out. These transportation requests are serviced by an inhomogeneous fleet of vehicles consisting of trucks and trucks with trailers. Both a truck and a trailer may carry at most one container.

The drivers of the vehicles are constrained in the time they can be on duty. In Germany due to contractual or legal obligations, a single driver may not be on duty for more than 8.5 hours. For long distance traffic, two drivers are needed. Their combined working time is 17 hours. In our application, we thus have four types of vehicles: trucks or trucks with a trailer both operated by one or two drivers. We shall refer to any one of these types by the term vehicle as long as a further distinction is not needed.

A route is a sequence of trips carried out by one vehicle. Routes may begin and end in different depots, although it is highly desireable to produce round trips. The majority of the transportation requests have to be carried out over night. The range of planning covers 24 hours. Three hours after a container has reached its final destination it will be unloaded and ready for balance tours.

If the parcel distribution company runs a hub-and-spoke type system container traffic between certain depots may not occur. Likewise, mail order services do not supply each other with goods. So the underlying network might be missing some arcs.

The following properties clearly distinguish our Container Transportation Problem (CTP) from other VRP described in the literature:

- Balancing the amount of empty containers.
  Mail order services, for example, deliver a huge amount of parcels every day. However, they do not receive any. In other words, loaded containers will not arrive at their depots. To ensure a sufficient number of empty containers to meet tomorrows need, empty containers have to be transported to these depots. In the language of OR, we have sources and sinks with respect to containers. The task is to find an appropriate set of trips to balance the number of containers in both sources and sinks. Therefore, the set of trips to be carried out is not completely known beforehand.
- For routes there is no TSP constraint.
  When combining trips to routes there exists no requirement to make it a shortest path route. Moreover, potential users strongly favour commuter traffic between nearby depots.

Some simplifications are necessary to cover the problem within the scope of this paper. We neglect mounting times and assume every vehicle to be operated by two drivers. A vehicle may carry 0, 1 or 2 containers.
As regards the cost function we model a simple case. We assume the whole fleet of vehicles to be rented for the purpose of transportation. Acquisition costs, taxes and so forth are expressed by a constant amount of money per vehicle used. Operational costs are assumed to simply be the product of a unit price per kilometer and the distance travelled. In some applications, they might also depend on the vehicle's home base.

3 MATHEMATICAL DESCRIPTION

Let $G = (V, A)$ be a directed and connected graph without loops. V is the set of depots, A is the set of arcs denoting the possible connections between the depots. With each arc $(i, j) \in A$ are associated nonnegative values d_{ij} denoting the distance between depot i and j, t_{ij} denoting the time needed to travel from depot i to j and k_{ij} denoting the costs. The corresponding matrices are $D = (d_{ij})$, $T = (t_{ij})$, and $K = (k_{ij})$. If no connection exists between the depots i and j we set $d_{ij} = d_{ji} = t_{ij} = t_{ji} = k_{ij} = k_{ji} = \infty$. The maximum time T two drivers can be on duty is seventeen hours in our case. If $t_{ij} < \infty$ then $t_{ij} \leq T$. The range of planning is a time interval $[0, P]$ with P normally being 24 hours.

Bearing in mind that in the CTP containers are to be transported we define a $trip\ c$ for a container to be a 5-tupel

$$c = (o, d, s, r, a)$$

It describes the task to move a container from the source depot $o \in V$ to the destination depot $d \in V$ along the arc $(o, d) \in A$. The status $s \in \{\mathsf{empty}, \mathsf{loaded}, \mathsf{nil}\}$ describes what kind of container to move. The first two states are self-explaning, the nil-state is needed to describe a trip without any container, also called an unproductive trip. Finally, $[r, a]$ is a time interval during which the trip must be carried out. At time r the container is ready for operation, at time a it should have arrived at its destination. Of course, the length of the interval must be sufficiently large, that is $r + t_{o,d} \le a$. For loaded containers, both r and a are given. An unproductive trip can be carried out anytime. In that case, we have $r = 0$ and $a = P$. For a trip with status empty the arrival time is always P since the container is needed the next day. A ready time, however, is not known in advance. There are times when in a certain depot at least one empty container is available and there are also time intervals when there is none. To overcome this difficulty the ready time is always assumed to be 0. With respect to the real life application this assumption is justified for two reasons. First, each depot has a certain number of empty containers in stock. Second, if a trip with an empty container is scheduled at a time when no one is available the depot manager will rent one from the free market. The side constraint in this case is to return the container at the end of the planning period. The rent for additional containers will increase the overall costs of a schedule only slightly so we can neglect them within the scope of this paper.

For a given trip c let o(c) denote its origin, d(c) its destination, r(c) its ready time and a(c) its arrival time, and s(c) its status.

A vehicle can be a truck with or without a trailer. In the vast majority of cases trucks with trailer will be used (cf. section 6). That is why we define a *route* ϱ of length n to be a (2, n)-matrix of trips

$$\begin{pmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,n} \\ c_{2,1} & c_{2,2} & \dots & c_{2,n} \end{pmatrix}$$

We shall refer to the trips $c_{1,i}$ as the trips for a truck and to $c_{2,i}$ as the trips for its trailer. If the trailer carries out unproductive trips only one can think of the vehicle as being a single truck.
Next, we introduce the auxiliary quantity

$$B_\varrho := \max_{i=1,\dots n} (0, \quad \max \quad (\mathtt{r}(c_{1,i}), \mathtt{r}(c_{2,i})) - \sum_{j=1}^{i} t_{\mathsf{o}(c_{1,j}),\mathsf{d}(c_{1,j})})$$

B_ϱ is the longest time interval between the ready time of a container at the i-th depot and the earliest possible arrival time of a vehicle at the i-th depot. So B_ϱ can be regarded as the earliest point in time when the execution of ϱ may begin.

For each route ϱ we require the following 5 conditions to be true:

$$\mathsf{o}(c_{i,j+1}) = \mathsf{d}(c_{i,j}), j = 1, \dots n-1; i = 1, 2 \quad (1)$$

$$\sum_{j=1}^{n} t_{\mathsf{o}(c_{i,j}),\mathsf{d}(c_{i,j})} \le M \quad i = 1, 2 \quad (2)$$

$$B_\varrho + \sum_{k=1}^{j-1} t_{\mathsf{o}(c_{i,k}),\mathsf{d}(c_{i,k})} \ge \mathtt{r}(c_j), j = 1, \dots n \quad (3)$$

$$B_\varrho + \sum_{k=1}^{j} t_{\mathsf{o}(c_{i,k}),\mathsf{d}(c_{i,k})} \le \mathsf{a}(c_j), j = 1, \dots n \quad (4)$$

$$\begin{array}{rcl} \mathsf{o}(c_{1,j}) & = & \mathsf{o}(c_{2,j}) \\ \mathsf{d}(c_{1,j}) & = & \mathsf{d}(c_{2,j}) \end{array} \quad j = 1, \dots n-1 \quad (5)$$

The meaning of these conditions is the following. We require a follow-up trip to start where its predecessor ends (1) and the tour length constraint to be fulfilled (2). According to the definition of B_ϱ (3) obviously holds and ensures a container is transported after it is ready. So for the vehicles no waiting times will occur. (4) guarantees that every container reaches its destination in time. Finally, we require the sequence of trips for the truck and its trailer to be parallel (5).

Let C be a set of trips. A set of routes is called a *schedule* for C iff every trip in C belongs to one and only one route. In real life, each route will be carried out by a vehicle. So the number of routes in a schedule equals the number of vehicles used.

Cost Function

For our application a medium-sized problem instance consists of approx. 3,000 containers. The program is able to handle up to 10,000 containers. Finding good solutions is important because, for example, the cost of operating large enough a fleet of vehicles is quite expensive. The enormous acquisition costs require managers to use the fleet to the fullest extent. However, operating costs require the vehicle routes to be as short as possible. Also, the acquisition costs of containers cannot simply be ignored.

The cost (fitness) function consists of two terms, one modelling acquisition costs and one modelling operating costs. Let Q be a constant amount of money representing acquisition cost per vehicle used. Executing a route ϱ costs

$$X(\varrho) := Q + \sum_{i=1}^{n} k_{\mathsf{o}(c_{1,i}),\mathsf{d}(c_{1,i})}$$

For a schedule C containing $||C||$ routes ϱ_i the cost function is

$$F := \sum_{i=1}^{||C||} X(\varrho_i)$$

Note that the cost function does not depend on the type of a vehicle, that is, whether a trailer is attached or not.

Optimization Goal

A certain number of loaded containers has to be transported. Any information needed to characterize these transportation requests in terms of trips is known in advance. So we are given a set L of trips representing loaded containers, i.e. $\mathsf{s}(\lambda) = \mathsf{loaded} \quad \forall \lambda \in L$. We also know the demand for empty containers. Provided the demand can be met by the number of containers currently available in the network we can construct a set of trips E describing balance tours, i.e. $\mathsf{s}(\varepsilon) = \mathsf{empty} \quad \forall \varepsilon \in E$. As a start, any method connecting sources and sinks will do. A certain problem instance is completely (but not uniquely!) described by the sets L and E.

Suppose we are given an instance of the CTP and the two sets E and L. Let N be a finite set of unproductive trips, i.e. $\mathsf{s}(\nu) = \mathsf{nil} \quad \forall \nu \in N$. A schedule for $L \cup E \cup N$ is called a *solution* of the CTP. Of course, we can add any finite number of unproductive trips to a schedule and still the requested transportation task will be performed. Our optimization goal now can be formulated as to find a least cost solution for the CTP.

4 GENETIC APPROACH

4.1 Encoding

Based on the definitions given above it is easy to find a genetic encoding. A gene corresponds to a route, a chromosome to a solution of the CTP. The number of genes is variable since the number of routes in a schedule is to be minimized. Each individual is made up of one chromosome.

Let S_1, S_2 be two schedules (individuals). Then $S_1 \cap S_2$ is the set of routes (genes) they have in common. We define the *normalized distance* of two schedules

$$\Delta(S_1, S_2) := \frac{||S_1|| + ||S_2|| - ||S_1 \cap S_2||}{||S_1|| + ||S_2||}$$

The measure Δ is 1 if the schedules have no route in common, and it is 0 if both schedules are identical. From Δ one can easily compute the mean distance between all individuals in a population, and also the genetic variance. For technical reasons, computing $S_1 \cap S_2$ is of order $O(||S_1|| \cdot ||S_2||)$. In order to achieve maximum efficiency, we assess the genetic variance of a population by Δ_{opt} as the mean normalized distance of the best individual to all selected parent individuals. As long as the best individual differs strongly from all other selected parents continuing the genetic search seems to be promising and Δ_{opt} will be comparatively high. If all individuals tend to become equal the selected parents will become more and more equal, too. So Δ_{opt} will be close to zero. A typical bar chart showing Δ_{opt} is shown in the lower half of Figure 1.

4.2 Initialization

From the optimization goal stated above follows that E might change during time. Moreover, every individual will have it's own set of balance tours. We will write $E_I(t)$ to denote the set of balance tours of the individual I at time t. At time $t = 0$, we assume $E_I(0) = E$ for each individual I.

Given the sets L and E for the current instance of the CTP one can easily find an initial solution. For every trip $\bar{c}_i \in L \cup E$ we create an unproductive trip

$$\bar{n}_i := (\mathsf{o}(\bar{c}), \mathsf{d}(\bar{c}), \mathsf{nil}, 0, P)$$

The set of all routes

$$\varrho_i := \begin{pmatrix} \bar{c}_i \\ \bar{n}_i \end{pmatrix}$$

is a schedule in which every trip is carried out by a single truck. Obviously, this schedule is a solution for the CTP.

The drawback, however, of this initialization process is that the genetic material of all individuals is identical. Therefore, we apply the Hill-Climber (cf. section 5) to each individual. As a result, the initial schedules differ sufficiently as can be seen from Figure 1. The normalized genetic distance Δ_{opt} shown in the lower half is about 0.3 immediately after initialization.

4.3 Mutation

An in-depth analysis of the CTP yields a few elementary strategies aiming at local improvements of a given solution. Two of them we will describe in more detail as they are the core strategies for the mutation operator. A third operation deals with balancing trips.
The mutation operates on a single individual I representing a solution S.

Attach a Trailer

Let $R \subseteq S$ be the set of routes where the trailers carry out unproductive trips only. If R is not empty select one route randomly, say ϱ. Remove the unproductive trips $c_{2,i}$, $i = 0, \ldots n$ for the trailer of ϱ from $L \cup E_I(t) \cup N$ and create

$$\varrho_T := \begin{pmatrix} c_{1,1} & c_{1,2} & \ldots & c_{1,n} \\ \bar{c}_{2,1} & \bar{c}_{2,2} & \ldots & \bar{c}_{2,n} \end{pmatrix}$$

The productive trips $\bar{c}_{2,i}$ are chosen randomly from $L \cup E_I(t)$ and removed from the routes they belonged to eventually breaking up those route into shorter routes. If ϱ_T is a feasible route it replaces ϱ.

There are two sides to this operation, a constructive one yielding a new route for ϱ now with a trailer, and a destructive one yielding a possibly non-empty set of short routes. To account for the latter, the elements in this set undergo a post-optimization process described in section 5.

Exchange Partial Routes

Let

$$\alpha := \begin{pmatrix} a_{1,1} & a_{1,2} & \ldots & a_{1,n} \\ a_{2,1} & a_{2,2} & \ldots & a_{2,n} \end{pmatrix}$$

and

$$\beta := \begin{pmatrix} b_{1,1} & b_{1,2} & \ldots & b_{1,m} \\ b_{2,1} & b_{2,2} & \ldots & b_{2,m} \end{pmatrix}$$

be two routes with a minimum length of 2. Assume there is a depot $x \in V$ with $\mathtt{d}(a_{1,i}) = x, 1 \leq i < n$ and $\mathtt{d}(b_{1,j}) = x, 1 \leq j < m$. If

$$\bar{\alpha} := \begin{pmatrix} a_{1,1} & \ldots & a_{1,i} & b_{1,j+1} & \ldots & b_{1,m} \\ a_{2,1} & \ldots & a_{2,i} & b_{2,j+1} & \ldots & b_{2,m} \end{pmatrix}$$

and

$$\bar{\beta} := \begin{pmatrix} b_{1,1} & \ldots & b_{1,j} & a_{1,i+1} & \ldots & a_{1,n} \\ b_{2,1} & \ldots & b_{2,j} & a_{2,i+1} & \ldots & a_{2,n)} \end{pmatrix}$$

are both feasible they replace α and β in S.

Varying Balancing Trips

Let ε be a trip describing the transport of an empty container. Remove ε from the route it belongs to perhaps breaking up this route into shorter ones. Next, choose a depot $x \in V$ at random and replace ε in $E_I(t)$ by two new trips, ε_1 going from $\mathtt{o}(\varepsilon)$ to x, ε_2 from x to $\mathtt{d}(\varepsilon)$. Similar to the initialization process for ε_1 and ε_2 two new routes are created where the trucks carry out ε_1 and ε_2, respectively. The trailers perform unproductive trips. Again, we end up with some routes of length 1 which are subject to a post-optimization process.

In fact, two complementary strategies operate on trips ε with $\mathtt{s}(\varepsilon) = nil$, the `Split` operation described above and a `Join` operation which is straightforward so we will not go into details about it.

4.4 Crossover

When designing the crossover operator the main idea came from the fact that the CTP is decomposable into similar subproblems. Splitting the network into two connected subgraphs there remains a CTP to be solved on either half plus a CTP for the transport crossing the border between the subgraphs. So the first step for the crossover operator is to define the two subgraphs. Let $x \in V$ be a depot and N the list of its neighboring depots, sorted by distance. In order to find a non-trivial decomposition choose a number i, $2 < i < \|N\|/2$, at random. N is the disjoint union of $N_1 := \{n_0, \ldots n_i\}$ and $N_2 := \{n_{i+1}, \ldots n_{\|V\|}\}$. Note that $n_0 = x$. Given two parent individuals p_1 and p_2 the offspring is constructed in the following steps:

1. Copy any route ϱ from p_1 into the offspring if ϱ visits depots in N_1 only.
2. Copy any route ϱ from p_2 into the offspring if ϱ visits depots in N_2 only.
3. Routes visiting depots in both N_1 and N_2 are copied if they exist in both p_1 and p_2.
4. Routes visiting depots in both N_1 and N_2 which exist in only one parent individual are split up into routes of length 1 and copied as in step 1 and 2. Similar to the mutation these split-up routes undergo a post-optimization process.

Due to the last step the offspring might contain routes not existing in either parent individual.

5 HILL-CLIMBING STRATEGIES

Many of the elementary optimizing strategies devised for the mutation operator can also be used in local search. While the mutation operator performs an operation in any case and will eventually adversely affect the overall fitness a Hill-Climber will perform an operation iff a local improvement is guaranteed. As mentioned in the previous section, a Hill-Climber is used as a post optimizer. The Hill-Climber operates on a single individual representing a solution S. For reasons of brevity, we give an informal description of two Hill-Climbing strategies only. The first one is an example for a strategy also used for mutation, the second is solely used in the Hill-Climber.

Attach a Trailer

Let $R \subseteq S$ be the set of routes where the trailers carry out unproductive trips only. If R is not empty select one route randomly, say ϱ. Find an element

$\bar{\varrho} \in R \setminus \{\varrho\}$ with productive trips $\bar{c}_{1,i}$, $i = 1 \ldots \bar{n}$ such that there exists a value k, $1 \leq k \leq n$ and

$$\begin{aligned} \mathtt{o}(c_{1,k+i}) &= \mathtt{o}(\bar{c}_{1,i}) \\ \mathtt{d}(c_{1,k+i}) &= \mathtt{d}(\bar{c}_{1,1}) \end{aligned} \qquad i = 1, \ldots \bar{n}$$

So we can replace at least some of the unproductive trips of ϱ's trailer by productive trips from $\bar{\varrho}$:

$$(c_{2,1} \ldots c_{2,k} \quad \bar{c}_{1,1} \ldots \bar{c}_{1,\bar{n}} \quad c_{2,k+\bar{n}} \ldots c_{2,n})$$

will be the new sequence of trips for ϱ's trailer. If ϱ is a feasible route it replaces $\bar{\varrho}$ in the schedule S.

Round Trips

Let $O \subseteq S$ be the set of routes of length 1. Select one at random, say ω. Select a depot $x \in V$, $x \neq \mathtt{o}(\omega)$ and $x \neq \mathtt{d}(\omega)$. With elements from O construct the set R of all feasible routes leading from $\mathtt{o}(\omega)$ to $\mathtt{d}(\omega)$ (which is ω itself), from there via x back to $\mathtt{o}(\omega)$. If R is not empty choose one element ϱ randomly, delete its constructing routes and add ϱ to the solution.

Replace unproductive trips

Let $O \subseteq S$ be the set of routes of length 1 where the trailer performs an unproductive trip. Select one at random, say ϱ. If there exists an unproductive trip u, $u \neq c_{2,1}$, with $\mathtt{o}(c_{1,1}) = \mathtt{o}(u)$ and $\mathtt{d}(c_{1,1}) = \mathtt{d}(u)$, replace u by $c_{1,1}$. If this operation yields a feasible route delete ϱ from the solution S.

In either case, no action will be performed if appropriate trips cannot be found. Thus, the Hill-Climber guarantees a local improvement in that it at least reduces the number of vehicles used.

6 EXPERIMENTS WITH THE BGA

We have set up a reference data set derived from real life data. It describes a problem instance with 45 depots, 2724 loaded containers and a need of 483 empty containers at various depots. All results we present in this paper were made using this reference data set. A run terminated after a certain number of generations g without improvement. With a population size p we have chosen the product $p \cdot g$ to be constantly 25,600. In other words, every population executes the same number of trials before ending the experiment. The experiments were carried out on a Pentium P166.

We like to point out that our CTP optimizer currently addresses the task of strategic planning. A master schedule is produced whenever the average freight volume has changed considerably. The daily schedules can then be derived. If only a few changes have to be made an experienced human planner can do this even manually. However, we recommend to obtain daily schedules by a special feature our optimizing software system offers. Describing this incremental optimization feature here is beyond the scope of this paper.

The optimizing program provides a variety of informations describing the resulting schedule in detail. Regrettably, some of these informations (prices, e.g.) are confidential. We restrict ourselves to two important data, the total number of kilometers driven by all vehicles and the fleet size.

For a better assessment of the results presented in this section we repeatedly set up experiments starting with all schedules produced so far as inital solutions. We finally ended up with a schedule containing routes for 1119 trucks with a trailer and another 34 trucks without trailer. The total distance travelled by this fleet is 511589 km. We shall refer to this schedule as the best solution found so far or the best solution, for short.

The CTP is implemented within the framework of a Breeder Genetic Algorithm as described in [Mühlenbein 1993]. For the genetic operators presented in the previous section the crucial point is to adjust their destructive and constructive components. The Response to Selection Equation [Mühlenbein 1994] gives a means to judge the performance of the breeding process. Here, we shortly summarize the derivation of this equation. For a deeper understanding the reader is referred to the original paper.

Let $M(t)$ denote the mean fitness of the population at time t. The change in fitness caused by selection

$$R(t) = M(t+1) - M(t)$$

is called the *response to selection*. $R(t)$ measures the expected progress of the population. The so called *selection differential* S is defined as the difference between the mean fitness of the selected parents $M_s(t)$ and the mean fitness of the population.

$$S(t) = M_s(t) - M(t)$$

In the process of artificial breeding both R and S can easily be computed. The connection between R and S is given by the equation

$$R(t) = b_t \cdot S(t)$$

In quantitative genetics b_t is called *realized heritability*. It is normally assumed that b_t is constant for a certain number of generations. This leads to

$$R(t) = b \cdot S(t)$$

Roughly speaking, the progress is due to a *genetic* search as long as the ration $R(t)/S(t)$ is greater than approx. 0.1. If it is approaching zero or becoming even less than zero *mutation* is the driving force. In Figure1[1] the curve in the upper half shows how $R(t)/S(t)$ develops during time. For our reference data set a population of only 32 individuals with truncation selection of 25% maintains a positive value of R/S for approx. 150 generations until for the first time a major impact from mutation occurs. At that time, the quality of the solution is about 5% worse than the best solution (expressed in DM). Larger populations maintain a positive value of $R(t)/S(t)$ for a considerably longer period of time. The curve in the lower half shows the normalized distance Δ_{opt} of the currently best solution to the rest of the selected parents.

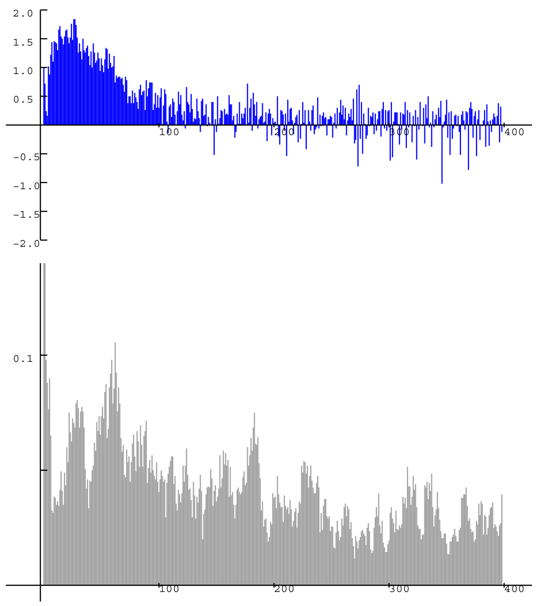

Figure 1: R(t)/S(t) (top)
Normalized distance Δ_{opt} (bottom)

Experiments with larger population sizes yield better results. This is a strong indication for a good exploration of the fitness landscape by the crossover operator. Table 1 shows the average results of 10 runs with the population size ranging from 32 to 128.

As regards total costs, the quality of all the schedules produced by a population of 128 individuals differ by approx. 0.6% at most.

For real life application the running times given in Table 1 seem prohibitive. But in practice the best

[1] The picture is taken directly from the graphical user interface of our CTP optimizer monitoring the process of breeding

PopSize	Average Values of		
	Km	Evals	Run Time
32	517692.8	7.644E+05	16.0h
64	515981.3	6.663E+05	10.5h
96	515251.9	7.190E+05	14.0h
128	513940.5	9.519E+05	19.5h

Table 1: PopSize vs. Quality

Dist. to	Average Values of		
best	Evals	StdDev	Run Time
5.0%	1.700E+04	1.241E+03	00:19
4.0%	2.670E+04	2.314E+03	00:29
3.0%	4.924E+04	4.984E+03	00:50
2.0%	1.419E+05	2.421E+04	02:30
1.5%	2.938E+05	6.756E+04	05:30
1.0%	6.238E+05	2.675E+05	12:00

Table 2: Convergence

solution is not needed. We know from potential customers they will accept any schedule within a 2% range from the best schedule we have found so far. The average convergence speed achieved in 10 runs with a population of 128 individuals is shown in Table 2. It takes about 50 minutes to generate a solution which is about 3% from the best schedule. After 2.5 hours the quality of the best individual is within the desired 2% range. This is acceptable for long range planning which will be the preferred range of application.

A benchmark with one of Germany's leading logistics companies showed that we are on the right track. Even in an early stage of development the quality of our schedules (expressed in km) is 10% to 15% better than what they can produce. Furthermore, our CTP optimizer is capable of dealing with much greater problem sizes and takes a lot more constraints and secondary optimization goals into account.

7 CONCLUSION

We have shown that a custom tailored genetic algorithm is able to solve a large scale container transportation problem. For our benchmark with approx. 3,200 transportation requests the memory size of one individual is about 300KB.

In this real world application many constraints have to be considered. Most of them have been integrated into the application specific mutation, crossover, and hill-climbing methods.

A real world benchmark of one of Germany's leading logistics companies has shown that the schedules produced by our optimizer are very good indeed.

Acknowledgement

This work was funded by the BMFT-Verbundprojekt HYBRID, subproject transport optimization under grant number 01 IB 405 A.

References

[Bodin 1981] Lawrence Bodin, Bruce Golden. *Classification in Vehicle Routing and Scheduling*. NETWORKS 11, 97-108 (1981)

[Desrochers 1990] M. Desrochers, J.K. Lenstra, M.W.P. Savelsbergh. *A classification scheme for vehicle routing and scheduling problems*. European Journal of Operational Research 46, 322-332 (1990)

[Mühlenbein 1992] H. Mühlenbein. *How Genetic Algorithms Really Work: Mutation and Hill-climbing* Parallel Problem Solving from Nature (PPSN II) Männer and Manderick (eds.), 15-26, North-Holland (1992)

[Mühlenbein 1993]
H. Mühlenbein, D. Schlierkamp-Voosen. *Predictive Models for the Breeder Genetic Algorithm*. Evolutionary Computation, 1(1): 25-49 (1993)

[Mühlenbein 1994]
H. Mühlenbein, D. Schlierkamp-Voosen. *The science of breeding and its application to the breeder genetic algorithm BGA* Evolutionary Computation, 1(4): 335-360 (1994)