Two Evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows

Jörg Homberger and Hermann Gehring

FernUniversität Hagen, Lehrstuhl Wirtschaftsinformatik, Profilstr. 8, D-58084 Hagen

Bundesrepublik Deutschland

Abstract:

The vehicle routing problem with time windows (VRPTW) is an extension of the well-known vehicle routing problem with a central depot. The objective is to design an optimal set of routes that services all customers and satisfies the given constraints, especially the time window constraints. The objective function considered here combines the minimization of the number of vehicles (primary criterion) and the total travel distance minimization (secondary criterion). In this paper, two evolution strategies for solving the VRPTW are proposed. The evolution strategies were tested on 58 problems from the literature with sizes varying from 100 to 417 customers and 2 to 54 vehicles. The generated new best known solutions indicate that evolution strategies are effective in reducing both the number of vehicles and the total travel distance.

Key words:

Evolution strategies, metaheuristics, vehicle routing, time windows.

1 Introduction and problem formulation

This paper considers the vehicle routing problem with time windows (VRPTW). The VRPTW is an extension of the well-known vehicle routing problem (VRP). It can be described as follows (see Domschke 1990):

n customers are to be serviced from a depot with vehicles of the same capacity Q. For each customer i, $i = 1,..., n$, there is a demand q_i, a service time s_i and a service time window $z_i = [e_i, f_i]$. The lower bound e_i describes the earliest and the upper bound f_i describes the latest time for start servicing. The demand q_i of a customer i is to be covered by exactly one service within the time window z_i. In addition, e_0 describes the earliest time for the departure of a vehicle from the depot i, $i = 0$, and f_0 the latest time for the arrival at the depot. The locations of the depot and the customers and the shortest distances $d_{i,j}$ and the corresponding travel times $d'_{i,j}$ between two locations are given. The objective is to determine a feasible route schedule which primarily minimizes the number of vehicles and secondarily the total travel distance.

Customers may not be serviced outside their time windows. However, time window constraints can still be met if a vehicle reaches a customer before the time window's lower bound. In this case the vehicle simply has to wait until the earliest time service can begin. Some variables are introduced for the subsequent consideration of time correlations: with equations (1) and (2) the earliest possible departure time δ_i, and with equations (3) and (4) the latest feasible arrival time α_i for the depot $i = 0$ or the customer i, $i \neq 0$ (see Solomon et al. 1988). The location $i-$ denotes here the predecessor of the location i in a route schedule and the location $i+$ describes the successor.

$$\delta_0 = e_0 \qquad \text{for the depot } i = 0, \tag{1}$$

$$\delta_i = \max \{\delta_{i-} + d'_{i-,i}, e_i\} + s_i \quad \text{for the customer } i,\ i \geq 1. \tag{2}$$

$\alpha_0 = f_0$ for the depot $i = 0$, (3)

$\alpha_i = \min \{\alpha_{i+} - d'_{i,i+} - s_i, f_i\}$ for the customer i, $i \geq 1$. (4)

According to Lenstra and Rinnooy Kan (1981) the VRP and the VRPTW belong to the class of the NP-hard combinatorial optimization problems. The VRPTW in particular, according to Solomon et al. (1988) is still "much more difficult" to solve than the VRP. It is therefore no surprise when primarily heuristic procedures are suggested for solving larger problem instances of the VRPTW. In the recent past, quite good results have been achieved for the VRP and VRPTW respectively, above all with metaheuristics such as tabu search (see Osman 1993, Taillard et al. 1996, Chiang and Russell 1997, Liu and Shen 1998), simulated annealing (see Chiang and Russell 1996) and genetic algorithms (see Thangiah et al. 1991). These metaheuristics have a common feature: they guide a subordinate heuristic in accordance with a concept derived from artificial intelligence, biology, mathematics, nature or physics to improve their performance (see Osman 1995). The group of metaheuristics also includes the evolution strategies (ES), since these strategies proceed in accordance with a concept borrowed from biological evolution and guide subordinate problem-related heuristics. The solution of the VRPTW by means of ES has not yet been reported in the literature. However, Ablay (1979) and Nissen (1994) apply ES to other combinatorial optimization problems.

ES were developed in the 1970s by Rechenberg (1973) and Schwefel (1977) to solve optimization problems with real-value variables. Especially the (μ, λ)-evolution strategy from Schwefel (1977) seems to be a particularly suitable method, since it evolves its so-called strategy parameters according to the ES metaphor. For this reason it will be used below to solve the VRPTW, after some fundamental characteristics of ES as well as a general metaheuristic based on the (μ, λ)-evolution strategy have been introduced. This general metaheuristic will be adapted to the VRPTW.

Finally, the resulting procedure will be described in detail and subjected to a comparative test. The test will be carried out with the problem instances published by Solomon (1987) and Russell (1995).

2 Evolution strategies as metaheuristics

2.1 Fundamentals of evolution strategies

Together with genetic algorithms (GA) and evolutionary programming (see Fogel 1992) evolution strategies form the class of evolutionary algorithms (see Nissen 1994). On the other hand, ES and GA also represent metaheuristics. Hence, they may also be described as evolutionary metaheuristics to distinguish them from other metaheuristics.

Similarly to GA, ES manipulate populations of individuals, which represent solutions of an optimization problem. Due to an integrated selection mechanism the iterative calculation of a sequence of populations favors the generation of better solutions. Differences to GA exist with regard to the representation of the problem and the search operators. ES dispense with the encoding of individuals and instead simulate the evolution process directly on the level of problem solutions. Accordingly, the search operators manipulate direct problem solutions and not coded individuals. In contrast to the GA, mutation operators are given a superior role in comparison to the recombination operators (see Gehring and Schütz 1994).

Originally, ES were developed to solve optimization problems with real-value decision variables. In the case of the optimization of an objective function $F(X)$ with n real-value decision variables x_i, $i = 1,..., n$, an individual can be represented as an n-dimensional real-value solution vector $X \in \mathbb{R}^n$. Various evolution strategies were developed to manipulate populations of real-value solution

vectors, e.g. the (1+1)-evolution strategy from Rechenberg (1973) and the (μ, λ)-evolution strategy from Schwefel (1977). The differences between these approaches concern primarily the population size, the representation of individuals and the search operators. The (1+1)-evolution strategy works with a population size of only a single individual, the (μ, λ)-evolution strategy with a population size of several individuals. In addition, the (1+1)-evolution strategy dispenses completely with a recombination operator and generates the offspring exclusively through mutation of real-value solution vectors. In contrast to the (1+1)-evolution strategy the (μ, λ)-evolution strategy uses an extended representation of individuals; in addition to the real-value solution vector each individual includes a second component, a real-value vector σ of the so-called strategy parameters. Both components are generally evolved by means of recombination and mutation operators. The following more detailed remarks are restricted to the (μ, λ)-evolution strategy.

Include Figure 1 here.

The (μ, λ)-evolution strategy is based on the reproduction process shown in Figure 1. Starting from a population $P(t)$ with μ individuals a number of λ, $\lambda > \mu$, offspring is generated. To calculate the offspring, several individuals, called parents, are selected from $P(t)$. Selections are random and put aside. The probability of selection is the same for each individual and independent of the fitness values $F^*(X)$ of the individuals. In the simplest case the fitness values $F^*(X)$ of the individuals correspond to the objective function values $F(X)$. Through recombination of the selected parents exactly one offspring is calculated and then subjected to a mutation. Finally, μ individuals of the following population $P(t+1)$ are selected from the set of λ offspring generated in this way. Now the fitness values $F^*(X)$ serve as the selection criterion; i.e. the μ individuals with the highest fitness values $F^*(X)$ are selected. Because the parents are not involved in the selection process, deteriorations during the search are permitted. Hence, the search may escape from bad local optima.

By means of the coefficient μ/λ the global character of the search, explorative or exploitative, can be influenced. A high value for this quotient leads to a low selection pressure. Accordingly, the individuals in the calculated following populations display greater diversity and the character of the search tends to be more explorative (see Hoffmeister and Bäck 1992).

The superior role of the mutation mechanism has already been mentioned. The solution vector X of an individual I is mutated through small random variations of X by means of a mutation rule. Critical is the choice of the mean size of the variations, the so-called mutation step size. A characteristic feature of the (μ, λ)-evolution strategy is the dynamization of the mutation step size in a way that enhances the success of the search. For this purpose the representation of an individual is extended by a vector σ of strategy parameters as follows:

$$I = (X, \sigma), \text{ with: } X = \{x_i | \, i = 1,..., n\}, \, \sigma = \{\sigma_i | \, i = 1,..., n\}. \tag{5}$$

Each element x_i of the real-value solution vector X is allocated exactly one element σ_i of the real-value vector σ of strategy parameters. In the given case, the strategy parameters σ_i, $i = 1,..., n$, represent standard deviations of normally distributed random variables with the expectation value 0. An individual is mutated by varying the elements of the solution vector according to the mutation rule in equation (6).

$$X' = X + NORM\,(\vec{0}, \sigma), \text{ with: } NORM\,(\vec{0}, \sigma) = \{NORM_i\,(0, \sigma_i) | \, i = 1,..., n\}. \tag{6}$$

For each element x_i of the solution vector X the value of the variation therefore represents a realization of the appropriate normal distribution $NORM_i\,(0, \sigma_i)$.

In the reproduction process, both parts, X and σ, of the respective parents are now inherited by the generated offspring. Along with the solutions, the values of the strategy parameters, which were

used to generate these solutions, are passed to the following population. This enables a twin-track learning process which tends to lead to better solutions and to more suitable values for the strategy parameters or mutation step sizes as well. According to the ES metaphor the vector σ of the strategy parameters is evolved through recombination of the strategy parameters of the respective parents.

2.2 A general evolutionary metaheuristic for combinatorial optimization problems

The described (μ, λ)-evolution strategy cannot be applied directly to combinatorial optimization problems since the discreteness of the solutions of combinatorial optimization problems is not considered. The individual components of the (μ, λ)-evolution strategy, namely

- the (μ, λ)-selection mechanism on the meta-level,
- the representation of individuals composed from solution vectors and strategy parameters and
- the local search in the solution space realized by means of subordinate heuristics, such as, for example, mutation,

are partly to be modified.

The strategy of the (μ, λ)-selection abstracts completely from the concrete representation of the problem, and for this reason can be taken over without any amendments.

Amendments are unavoidable with the representation of individuals. The solution vector X consists of n discrete elements. In the following we abbreviate a discrete solution vector to *S*, *SVector* or *SV*. Depending on the problem, it may be practical to divide the vector elements into groups. The previous form of the vector σ of the strategy parameters cannot be retained, but the idea behind it of mutation and adaptation of the mutation step size can be retained. The vector σ is now replaced by

information which "encodes" the transition from a given solution S to a solution S' in the neighbourhood $N(S)$ of S. Since two optimization criteria must be taken into account, additional information is to be taken over in the representation of the individuals concerning the search direction. When a new solution $S' \in N(S)$ is generated, the weight may then be placed more on the first or on the second optimization criterion.

The heuristic used on the level of the local search covers two conceptual elements, a neighbourhood concept and a rule for generating neighbourhood solutions. The neighbourhood $N(S)$ of a solution S consists here of the set of feasible solutions that can be generated from S by means of a rule. The rule for generating neighbourhood solutions can therefore be conceived as a "move-generation mechanism" according to Osman (1995). Osman states:

"A move-generation mechanism generates the set of neighbours changing one attribute or a combination of attributes of a given instance S. A move-generation is a transition from a solution S to another solution $S' \in N(S)$ in one step (or iteration)".

If a move defined in this way effects only relatively minor amendments of a given solution S, it can be understood as a mutation operator. A simple concept for varying the mutation step size is now obvious: when a new solution S' is generated either only a single move is carried out or several moves.

A modification of the neighbourhood concept would be necessary if a "solution-generation mechanism" based on Osman (1995) were used to generate a solution, because this mechanism works on several solutions:

"A solution-generation mechanism works on a set of solutions rather than a set of attributes for a single solution".

Here, the "solution-generation mechanism" is not used to generate new solution vectors, but it is used to adapt the mutation rule or the "move-generation mechanism". This is always possible if a suitable representation form, for example as a sequence vector, can be found for a move. In this case, the different crossover operators described in the literature can be used for the recombination of the mutation codes of two individuals. For example, the cycle crossover from Oliver et al. (1987) or the uniform order-based crossover from Davis (1991).

Taken together, this results in the concept of an evolutionary metaheuristic for combinatorial optimization problems illustrated in Figure 2.

Include Figure 2 here.

On the meta-level, the sequence of generated populations is controlled by means of the (μ, λ)-selection. As before, the global character of the search can be varied through the quotient μ/λ. In the same way, a twin-track learning process aimed at the improvement of the solutions is also enabled: On the one hand, the mutation of individuals in combination with the (μ, λ)-selection tends to lead to better solutions; on the other hand, the recombination of the mutation codes of individuals, again in combination with the (μ, λ)-selection, tends to lead to more suitable realizations of the mutation codes.

A simplified variant of the outlined metaheuristic results if the recombination of the mutation codes is dispensed with. In this case, there is only a single-track learning process. On the other hand, more complex variants are also conceivable. For example, the mutation code generated through

recombination may be subjected additionally to a mutation. Only two variants are dealt with below, the version shown in Figure 2 and the simplified version.

3 Adaptation of the general evolutionary metaheuristic to the VRPTW

In the following two sections the two versions of the general evolutionary metaheuristic are adapted to the VRPTW: the simplified version in Section 3.1, and in Section 3.2 the version shown in Figure 2. The adaptation concerns primarily the representation of individuals and the design of the mutation operators and of the recombination operator. The procedural components concerning the evaluation of individuals and a refining of the (μ, λ)-selection are the same in both cases. For this reason they are treated separately in Section 3.3. Finally, the configuration of the two evolution strategies is described in Section 3.4.

3.1 Evolution strategy ES1

The recombination of the mutation codes is not included in evolution strategy ES1. Hence, a new individual is generated as illustrated in Figure 3. In the following the representation of an individual is first stated more precisely, and then the mutation rule.

Include Figure 3 here.

In Figure 3 *SVector* denotes a feasible solution of the VRPTW, i.e. a feasible route schedule. As demonstrated in Figure 4, a route schedule can be represented in the form of a discrete solution vector. For reasons of clarity, the connections with the depot of the first and final customers are omitted in Figure 4a).

Include Figure 4 here.

The strategy-orientated parameters *NMoves* and *RElimination* serve the following purposes:

- *NMoves* indicates how often the move operator used is carried out during a mutation.

- The binary parameter *RElimination* $\in \{0, 1\}$ is used to code the search direction within a mutation step. The value 0 denotes the pure travel distance minimization within a mutation step and the value 1 denotes a two-phase process in which, subsequently to an undirected search step, an attempt is made to minimize the number of vehicles.

This means that the parameter *NMoves* sets only the mutation step size and has no further influence on the sequence of a mutation. As Figure 3 shows, the values of *NMoves* and *RElimination* are passed on without variation from one individual to a new individual. Multiple experiments, in which the strategy parameters where evolved concurrently with the solutions, delivered no better results.

In order to reproduce an individual its route schedule *SVector* is varied by means of a mutation rule. The result is a feasible route schedule *SVector'* $\in$ *N*(*SVector*) selected randomly from the neighbourhood *N*(*SVector*). With regard to the choice of a suitable neighbourhood structure, the more complex concept of the "compound move" is used. Compound moves consist of different simple move operators. According to Glover (1991) and Osman (1993) they represent a particularly effective concept. In the given case, two simple move operators are integrated in a compound move:

- a move operator randomly selected from the set *MoveSet* = {Or-opt move, 2-opt* move, 1-interchange move};

- a modified Or-opt move operator, which serves the aim of reducing the number of vehicles through elimination of a route.

However, this compound move is only applied to special individuals, in the case of the other individuals the compound move is reduced to a simple move. For details see Figure 5, in which the complete mutation rule is described.

Include Figure 5 here.

As Figure 5 shows, a compound move is only carried out if the mutated individual explicitly demands this. A compound move only leads to success if it is actually possible to eliminate the route. In other case, the number of routes is not changed. Restricting the compound move to a subset of individuals is meaningful, because eliminating a route often leads to an increase in the total travel distance.

Some remarks on the move operators are necessary to define the mutation rule even more closely.

The operators of the set *MoveSet* are due to the following authors: The Or-opt move is based on an exchange concept of Or (1976). The 1-interchange move was introduced by Osman (1993) and the 2-opt* move by Potvin and Rousseau (1995). In the following the Or-opt move will be described briefly as an example.

An Or-opt move consists of a removal and an insertion. With these operations a customer or a sequence of customers is removed from a route R in a route schedule *SVector* and then inserted at another position in *SVector*. Here, the customers to be removed as well as the insertion position are determined at random. If there is only one customer the procedure is as follows:

- A customer k is removed from the route R, $R = (0,..., k–, k, k+,..., 0)$, by serving the remaining customers in route R in the original sequence with the exclusion of customer k. The resulting route R_1, now reduced by customer k, reads: $R_1 = (0,..., k–, k+,..., 0)$. If route R is a single route,

i.e. the vehicle serves customer *k* only, customer *k* is removed from route *R* by eliminating the route.

- A removed customer *k* is inserted into a route R_2 of the route schedule by extending the set of customers being served in route R_2. If the insertion position is denoted by two consecutive locations (*i*, *i*+), route R_2 reads: R_2 = (0,..., *i*, *i*+,..., 0). The insertion of customer *k* results in route R_3 = (0,..., *i*, *k*, *i*+,..., 0). In principle, it is not impossible that customer *k* is placed in the route from which he was removed. However, the insertion position may differ from the removal position.

According to Figure 5, the modified Or-opt operator tries to compute a route schedule with a reduced number of vehicles. For this purpose an attempt is made to eliminate the route $R \in SVector'$ in the current route schedule *SVector'* which has the lowest number of customers, in the following denoted as "smallest" route. Hence, customer insertions are defined more precisely as follows: the customers of the smallest route *R* are inserted in accordance with their service sequence consecutively in a route $R' \neq R$, $R' \in SVector'$, – provided the constraints of the VRPTW are not violated. If there exist alternative insertion positions (*i*, *i*+) in other routes for a customer $k \in R$, two cases must be differentiated:

- If an index *i*+ denotes a customer, the shift in the earliest possible departure time for customer *i*+ resulting from the insertion is used as decision criterion.

- On the other hand, if an index *i*+ denotes the depot, the shift in the earliest possible arrival time at the depot *i*+ serves as decision criterion.

The insertion position with the shortest shift in time $PT_k(i, i+)$ is selected:

$$PT_k(i,i+) = \begin{cases} \max\{\max\{\delta_i + d'_{i,k},\, e_k\} + s_k + d'_{k,i+},\, e_{i+}\} + s_{i+} - \delta_{i+} & \text{if} \quad i+ > 0, \\ \max\{\delta_i + d'_{i,k},\, e_k\} + s_k + d'_{k,0} - (\delta_i + d'_{i,0}) & \text{otherwise}. \end{cases} \tag{7}$$

If there exist several feasible insertion positions with the same minimal shift times, the insertion position is selected for which the additional travel distance $PD_k(i, i+)$ is a minimum. $PD_k(i, i+)$ is determined as follows:

$$PD_k(i, i+) = d_{i,k} + d_{k,i+} - d_{i,i+}. \tag{8}$$

If no customer in route R can be inserted in another route $R' \neq R$, $R' \in SVector'$ without violating the constraints then the route schedule *SVector'* remains unchanged.

The search behavior of the evolution strategy ES1 can be changed by varying the strategy parameters *NMoves* and *RElimination*. Higher values for the mutation step size favor escapes from bad local optima, whereas lower values lead to a more intensive local search. An intensification is indicated, for example, if – for a fixed number of vehicles – the total travel distance is to be minimized. Greater values of *NMoves* will therefore be connected with the aim of minimizing the number of vehicles (*RElimination* = 1), and lower values with the aim of minimizing the total travel distance (*RElimination* = 0).

3.2 Evolution strategy ES2

Evolution strategy ES2 is based on the concept of an evolutionary metaheuristic shown in Figure 2. This concept has already been explained and will be adapted below to the VRPTW. The interplay between recombination and mutation, and the design of the recombination process and the mutation rule are in the foreground.

Include Figure 6 here.

Figure 6 shows how recombination and mutation interlock. Starting from three parents, an offspring is generated in two consecutive steps. The temporary offspring found after the first step establishes the connection between both steps.

In the first step, a temporary offspring is generated through

- recombination of the mutation codes of two parents and take over of the resulting mutation code MC'_3 in the temporary offspring;
- copying the solution vector SV_3 and the mutation direction RE_3 of a third parent into the temporary offspring.

The generation of the mutation code MC'_3 of the temporary offspring imitates the principle of sexual reproduction. In contrast, the subsequent copying procedure is based on the principle of biological replication, i.e. doubling of genetic material. After the recombination no progress has been achieved in the solution space. The generated temporary offspring represents the same solution as the third parent.

For the purpose of recombining two mutation codes proven crossover operators from the area of genetic algorithms may be used. Here, the uniform order-based crossover, proposed by Davis (1991), was selected. This operator is tailored to codes that represent permutations or sequences.

In the second step, the temporary offspring is transferred to the definite offspring by subjecting the inherited route schedule SV_3 to a mutation. In this case, a mutation covers a set of removal and insertion operations whose sequence is controlled through the mutation code. In contrast to the evolution strategy ES1 the mutation code consists here of a so-called sequence vector of length $2n$.

The elements of the sequence vector represent customer numbers; each customer number occurs exactly twice in a sequence vector. The first occurrence of a customer number identifies a removal operation and the second an insertion operation. The following applies to the removal or insertion of a customer:

- The removal of a customer is performed exclusively with the removal operation of the Or-opt operator.

- The insertion of a customer is based on the following insertion heuristic: For each route $R \in SV_3$ of route schedule SV_3 it is proved whether an extension of the route R by inserting customer k is feasible. If an extension of one or more routes by inserting customer k is feasible, the insertion position is selected so that the additional travel distance according to equation (8) is minimal. If the insertion of customer k into each of the routes violates the given constraints, a single route serving customer k is added to the route schedule.

Eventually, the removal and insertion operations are followed by an additional operation:

- If the parameter *RElimination* of the generated offspring has the value 1, a modified Or-opt move is carried out in the same way as with evolution strategy ES1. Here as well the target is to reduce the number of vehicles by eliminating a route.

Include Figure 7 here.

Figure 7 illustrates the control of removal and insertion operations with a sequence vector. The vector is processed from left to right. At each position it is checked whether a customer number occurs for the first or second time. In the first case a removal operation is carried out and in the second an insertion operation. In Figure 7 the first occurrence of the customer numbers 8, 4 and 3 causes the removal of these customers from the route schedule. Subsequently, the second

occurrence of customer number 4 causes the insertion of this customer in the route schedule. Finally, customers 1, 8 and 9 are inserted in the route schedule.

The removal and insertion operations carried out according to a sequence vector can neutralize each other's effects to different degrees. Depending on the extent of this compensation, variations to the route schedule or the mutation steps with different sizes occur randomly. In the course of the algorithm, however, the convergence of the mutation codes leads to reduced mutation step sizes. Initially, therefore, the mutation step sizes are greater and towards the end of the search they are smaller.

3.3 Evaluation and selection of individuals

As to the evaluation and selection of individuals in the reproduction process, the above arguments result in the following concept: from λ, $\lambda > 0$, generated individuals, the μ, $\mu < \lambda$, individuals with the highest fitness values *F*(SVector)* are inserted into the following population. Here, *F*(SVector)* represents a lexicographic order relation which takes into account the number of vehicles with first priority and the total travel distance with second priority. However, a test of this concept led to unsatisfactory results for two reasons:

- If, for a given number of vehicles, the further search direction is determined exclusively through the secondary criterion, i.e. minimization of the total travel distance, the achievement of the primary criterion, i.e. minimization of the number of vehicles, may be impeded. Retzko (1995) shows that usually the total travel distance increases if the number of vehicles falls below a defined value. Vice versa it appears plausible that minimization of the total travel distance does not inevitably lead to a reduction in the number of vehicles.

- The given neighbourhood structure seems not to favor the achievement of the primary optimization criterion. On the one hand, new solutions are generated by subjecting existing individuals to relatively minor variations. On the other hand, a route usually serves a larger number of customers. Hence, the generation of a new route schedule will very rarely lead to a reduction in the number of vehicles. This applies all the more as the time window constraints render it more difficult to distribute the customers over a smaller number of routes.

Therefore, the question arises how the search can be guided in a direction reducing the number of vehicles. For this purpose, a third evaluation criterion is introduced, which consists of two indices. Both indices refer to the smallest route R of a route schedule *SVector* and estimate in different ways how easily route R can be eliminated in the course of the further search.

The first, simple index is the number C_R of the customers served in route R. It is assumed that a lower value C_R favors the elimination of route R.

The second index is the so-called "minimal delay" D_R for all customers on route R. The property D_R results from the addition of appropriate customer-related values D_k over all customers k, $k \neq 0$, of route R:

$$D_R = \sum_{k \in R} D_k \, . \tag{9}$$

The property D_k denotes the "minimal delay" caused by inserting customer $k \in R$ into another route R', $R' \neq R$, of the route schedule.

When a customer k is inserted between two consecutive locations i and $i+$ in a route R', three cases must be differentiated:

(1) There exist several other routes that can be extended by the service for customer k without a violation of the relevant constraints. In this case there is no delay and $D_k = 0$.

(2) There is no other route that can be extended by the service for customer k without a violation of the appropriate vehicle capacity constraint. In this case, customer k cannot be inserted into another route and D_k is set to infinite: $D_k = \infty$.

(3) There exist one or more routes that can be extended by the service for customer k under the sole violation of time window constraints. In this case, the so-called "delay" $D_k(i, i+)$ is calculated for all pairs of consecutive locations $(i, i+)$ between which the customer k can be inserted. $D_k(i, i+)$ is composed by adding the two time shifts $V1_k(i, i+)$ and $V2_k(i, i+)$. $V1_k(i, i+)$ indicates how many time units the time window's upper bound f_k for the inserted customer k is delayed through the insertion of customer k in route R'. And $V2_k(i, i+)$ indicates by how many time units

- the latest feasible arrival time α_{i+} for customer $i+$, if $i+$ denotes a customer, or
- the latest feasible arrival time α_0 at the depot, if $i+$ denotes the depot,

is delayed through the insertion of customer k in route R'. In detail:

$$D_k(i, i+) = V1_k(i, i+) + V2_k(i, i+), \text{ whereby:} \quad (10)$$

$$V1_k(i, i+) = \max\{\delta_i + d'_{i,k} - f_k, 0\}, \quad (11)$$

$$V2_k(i, i+) = \max\{\max\{\delta_i + d'_{i,k}, e_k\} + s_k + d'_{k,i+} - \alpha_{i+}, 0\}. \quad (12)$$

The formulae for computing $V1_k(i, i+)$ and $V2_k(i, i+)$ are illustrated by the example shown in Figure 8. In this example, customer $k = 4$ is inserted between locations $i = 2$ and $i+ = 3$.

Include Figure 8 here.

After customers 1 and 2 were serviced within their time window, the vehicle reaches the inserted customer $k = 4$ outside the service time window. Service for the customer is delayed, in

relation to the latest feasible start of service, by the forward shift in time $V1_4(2, 3)$ shown. By means of backward calculation starting from the depot the latest feasible arrival time α_3 at customer $i+ = 3$ can be calculated for the case that the vehicle returns to the depot just in time. However, if the vehicle returns after time α_3, as is assumed in Figure 8, there is a positive time difference, the forward shift in time $V2_4(2, 3)$ also shown in Figure 8.

Taking this differentiation into account, the rule for determining the "minimal delay" D_k for a customer $k \in R$ can be given. Let A_k denote the set of all pairs of locations $(i, i+)$ which have to be considered in case (3), then:

$$D_k = \begin{cases} 0 & \text{for the case (1),} \\ \infty & \text{for the case (2),} \\ \min\limits_{(i,i+)\in A_k} D_k(i, i+) & \text{for the case (3).} \end{cases} \qquad (13)$$

The index D_R is only a very rough estimate with regard to the question how easily route R can be eliminated. On the one hand, interdependencies of time, which may arise when all customers from route R are inserted simultaneously into other routes, are not taken into consideration. On the other hand, the determination of property D_k is based on an optimistic concept. A pessimistic concept would also be plausible, or an expectation concept.

In the third evaluation criterion, which is designated below as the "terminability of the smallest route", the two indices C_R and D_R are considered in the sense of a lexicographic order relation. The evaluation of an individual is the higher, the smaller the number of customers C_R is, and, in the case of individuals with the same number of customers, the lower the value D_R is.

For the inclusion of the third evaluation criterion in the evaluation rule *F*(SVector)* there exist several alternatives. For example, the previous lexicographic order relation can be extended by considering the terminability of the smallest route with third priority. However, more success is promised by a procedure in two phases with different search directions. Kursawe (1992) reports on the successful application of a multiphase (μ, λ)-selection in multicriterion optimization with evolution strategies. A two-phase concept for evaluation and selection was realized in conformity with Kursawe:

- In the first phase, the generated λ individuals of a generation are transformed to a lexicographic order which takes into consideration the number of vehicles with first priority and the terminability of the smallest route with second priority. The κ, $\kappa < \mu$, individuals with the best evaluation are taken into the following population.

- In the second phase, once again all λ individuals of a generation are transformed to a lexicographic order. After the number of vehicles, however, the total travel distance is now taken into consideration with second priority. The $\mu - \kappa$ individuals with the best evaluation are taken into the following population.

Depending on the choice of the quotient κ/μ the minimization of the number of vehicles is emphasized more or less in comparison to the minimization of the total travel distance.

3.4 Configuration of the evolution strategies ES1 and ES2

The configuration of the evolution strategies ES1 and ES2 concerns the initialization, the (μ, λ)-selection and the termination. These elements were specified as follows:

(1) Initialization of the starting population: the individuals of a starting population were generated both for ES1 and for ES2 by means of a stochastic approach which was based on the savings algorithm from Clarke and Wright (1964). In this approach, the stochastic element consists of the random selection of savings elements from the savings list. For both evolution strategies, ES1 and ES2, the values of the parameter *RElimination* are determined randomly, and in the case of strategy ES2 the mutation codes of the individuals in the starting population were also determined as random sequence vectors. Finally, for evolution strategy ES1 the mutation code, i.e. the mutation step size *NMoves*, for each individual in the starting population was chosen randomly from the interval [1,..., 10]. As tests have shown, evolution strategy ES1 reacts relatively robustly to changes of the upper bound for this parameter. However, values which were selected too low for *NMoves* frequently led to premature convergence against local optima.

(2) (μ, λ)-selection and population size: the values chosen for λ, μ and κ are given in Table 1. Here a suggestion from Schwefel (1987) was taken into account which recommends a value of $\mu/\lambda \approx 1/7$ for the quotient μ/λ. The reason for selecting relatively large parameter values in ES2 is as follows: low values of the population size favor a premature convergence of the sequence vectors with the consequence that the search ends rapidly in local optima.

Include Table 1 here.

(3) Termination condition: all computations were terminated after a given time limit *TL* = 1800 seconds. However, at a much earlier time solutions were usually found which could not be improved with regard to the number of vehicles and the total travel distance; the respective computation times are reported in Section 4.2.

4. Evaluation

4.1 Test problems

In order to demonstrate the solution behavior of the evolution strategies for different problem sizes, two groups of problem instances were selected. These groups differ in the number of customers per instance.

The first group consists of the 56 problem instances described by Solomon (1987). Each instance of this group comprises 100 customers. The location of the customers and the depot are given in a Cartesian coordinate system. The location coordinates are integer values between 0 and 100.

The second group comprises the two instances D-417 and E-417 given by Russell (1995) which are both taken from practice. Both instances comprise 417 customers. The customer locations and the depot are also given in a Cartesian coordinate system and the location coordinates are again integer values.

In accordance with the procedure usually found in the literature, the Euclidean distances $d_{i,j}$ and travel times $d'_{i,j}$ were calculated exactly to two decimal places. To verify the feasibility and accuracy, final results were recalculated with the greatest degree of accuracy fixed by the computer.

4.2 Computational results

Optimal solutions are known only for some simple problem instances. For this reason, it seems obvious to evaluate methods on the basis of best solutions. The results obtained with ES1 and ES2 will be contrasted with results published in the literature or in the Internet. All calculations were

carried out on a PC (Pentium processor, 200 MHz). ES1 and ES2 were implemented in C and compiled with Borland 5.02.

Each of the 56 problem instances from Solomon was calculated 10 times with method ES1 and method ES2, respectively. Each calculation of a problem instance was based on a starting population generated at random and a set of constant parameters as described in Section 3.4. In the following, the achieved results are presented on three levels: averages over each class of problems (C1, C2, R1, R2, RC1, RC2), best achieved solution for each of the 56 problem instances and complete solutions with the sequence of customers for certain instances.

Averages over each problem class are given in Table 2. In this table, *MNV* denotes the mean number of vehicles, *MND* the mean travel distance and *MCT* the mean computing time. The calculation of *MNV*, *MND* and *MCT* is always based on the best solutions that were achieved for the problem instances of the respective problem class. In the bottom line, the cumulated number of vehicles *CNV* and the cumulated total travel distance *CND* are shown. A direct comparison of the results reported in Table 2 is difficult, because they have been obtained on different computer plattforms. Furthermore, the number of independent runs used to calculate the averages differs from author to author.

Insert Table 2 here.

The results of the comparative test presented in Table 2 may be summarised as follows:

- For problem class R1 both methods, ES1 and ES2, lead to new best means *MNV*; for the remaining problem classes, the best known mean *MNV* is also achieved either with both methods (see C1, C2, R2, RC2) or with one of them (see RC1).

- The mean travel distances *MND* calculated for ES1 and ES2 are partly equivalent to the best known mean values (see C1, C2) and partly (considerably) higher (see R1, R2, RC1, RC2).

- The cumulated number of vehicles *CNV* amounts to 406 for ES1 and ES2; in comparison with the other methods this value is the lowest. However, the cumulated total travel distance *CND* for ES1 and ES2 is higher than the respective values for some other methods.

With respect to the primary optimization criterion, ES1 and ES2 tend to outperform previous methods. This tendency is definitely not caused by a general superiority of the evolution strategy over other metaheuristics such as genetic algorithms, tabu search and simulated annealing. It rather has to be assumed that developed evaluation function and neighbourhood operators explain the above finding. These methodical components consider explicitly the primary optimization criterion and initiate repeated attempts to reduce the number of vehicles. Thus, the generation of neighbourhood solutions with lower vehicle numbers is favoured.

A comparison of the best achieved solutions for each of the 56 problem instances is shown in Table 3. The results may be summarised as follows:

- For one instance (R101) ES1 and ES2 calculate solutions with one vehicle more than the best known solution, and for two instances (R104, R112) ES1 and ES2 calculate solutions with one vehicle less than the previous best solutions.

- For 51 of the remaining 53 instances the vehicle numbers *NV* determined by ES1 and ES2 are equal to the respective *NV*-value of the best published solutions, and for two instances either ES1 (see R109) or ES2 (see RC106) found a solution with the same vehicle number as reported for the best published solution.

- In the case of equal vehicle numbers *NV* the quality of compared solutions depends on the respective travel distances *ND*. Here, in 8 of the above-mentioned 53 relevant cases ES1 achieved the lowest value of *ND*, in 9 cases ES2 and in 18 cases other methods (see column „Reference“ in Table 3); in the remaining 18 cases the compared travel distances *ND* are (supposed to be) equal.

Insert Table 3 here.

The performance of ES1 and ES2 relative to the seven known optimal solutions to the Solomon test problems is also of interest. The seven problem instances in question are R101, R102, C101, C102, C106, C107, and C108. For R101, ES1 and ES2 calculate solutions with one vehicle more than the optimal solution and for R102 the achieved vehicle numbers are equal to the vehicle number of the optimal solution. In the case of the remaining five clustered problem instances the vehicle numbers and the travel distances calculated by ES1 and ES2 are equal to the respective values of the optimal solutions; the differences occurring between the travel distances are caused by different computational accuracies (truncation to one decimal place versus greatest degree of accuracy fixed by the computer).

It should be noted that each of the methods included in the comparison uses the minimization of the vehicle number as primary optimization criterion. The methods of Potvin and Bengio (1996), Potvin et al. (1996), Russell (1995), Chiang and Russell (1996) consider the total route time as their second priority. The remaining methods, as well as ES1 and ES2, use the minimization of the total travel distance as the secondary objective criterion.

Three complete solutions that include the sequence of customers are described in Table 4. The respective problem instances are R104, R112, and RC202. As already mentioned, the solutions for R104 and R112 comprise one vehicle less than the previous best solutions. RC202 is an

representative example of the group of those 17 problem instances, for which either ES1 or ES2 determines a new best solution with respect to the total travel distance only.

Insert Table 4 here.

Further, Table 5 presents corresponding computing results for the two instances D-417 and E-417 from Russell (1995). Each instance was calculated only 5 times with both method ES1 and method ES2. As Table 5 shows, only those best solutions reported by Liu and Shen (1998) are slightly better than the best solutions generated with ES1. Due to the small experimental basis, this observation may only be interpreted as a weak tendency.

Insert Table 5 here.

In summary it can be stated that the developed evolution strategies are certainly on a par with other heuristics for solving the VRPTW. Other combinatorial optimization problems can probably also be solved successfully with evolution strategies. Finally, it should be noted that the proposed metaheuristics are generalizable to extended VRPTW models, such as those requiring different fleet vehicle capacities Q, mandated driver break time, etc. Additional restrictions of this kind may always be considered when a new individual is generated by means of a mutation rule.

Acknowledgement

The authors are highly indebted to three anonymous referees for their very helpful and constructive comments.

References

Ablay, P. (1979) Optimieren mit Evolutionsstrategien. Doktorarbeit, Fachbereich Wirtschafts- und Sozialwissenschaften, Universität Heidelberg, Heidelberg.

Bachem, A., Bodmann, M., Bolz, G., Emden-Weinert, T., Erdmann, A., Kiahaschemi, M., Monien, B., Prömel, H. J., Schepers, J., Schrader, R., Schulze, J. and Tschöke, S. (1997) Verbund-projekt PARALOR: Parallele Algorithmen zur Wegeoptimierung in Flugplanung und Logistik. Report No. 97-269, Beitrag zur Statustagung des BMBF HPSC 97, Angewandte Mathematik und Informatik, Universität Köln, Köln.

Chiang, W.-C. and Russell, R. A. (1996) Simulated annealing metaheuristics for the vehicle routing problem with time windows. *Annals of Operations Research*, **63**, 3-27.

Chiang, W.-C. and Russell, R. A. (1997) A reactive tabu search metaheuristic for the vehicle routing problem with time windows. *INFORMS Journal on Computing* **9**, 417-430.

Clarke, G. and Wright, J. W. (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, **12**, 568-581.

Davis, L. (1991) *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, New York.

Desrochers, M., Desrosiers, J. and Solomon, M. M. (1992) A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research* **40**, 342-354.

Domschke, W. (1990) *Logistik: Rundreisen und Touren*. Oldenbourg, München.

Fogel, D. B. (1992) Evolving artificial intelligence. Ph.D. thesis, University of California, San Diego.

Gehring, H. and Schütz, G. (1994) Zwei genetische Algorithmen zur Lösung des Bandabgleich-problems. In *Operations Research: Reflexionen aus Theorie und Praxis*, ed. B. Werners and R. Gabriel, pp. 85-128. Springer, Berlin.

Glover, F. (1991) Multilevel tabu search and embedded search neighbourhoods for the traveling salesman problem. Working paper, Graduate School of Business and Administration, University of Colorado, Boulder, Colorado.

Hoffmeister, F. and Bäck, T. (1992) Genetic algorithms and evolution strategies: similarities and differences. Technical report SYS-1/92, Fachbereich Informatik, Universität Dortmund, Dortmund.

Kontoravdis, G. and Bard, J. F. (1995) A GRASP for the vehicle routing problem with time windows. *ORSA Journal on Computing*, **7**, 10-23.

Kursawe, F. (1991) A variant of evolution strategies for vector optimization. In *Parallel Problem Solving from Nature*, ed. H.-P. Schwefel and R. Männer, pp. 193-197. Springer, Berlin.

Lenstra, J. and Rinnooy Kan, A. (1981) Complexity of vehicle routing and scheduling problems. *Networks*, **11**, 221-227.

Liu, F. and Shen, S. (1998) A route-neighbourhood-based metaheuristic for vehicle routing problem with time windows. Working paper, National Chiao University, Hsinchu, Taiwan.

Nissen, V. (1994) *Evolutionäre Algorithmen*. Deutscher Universitäts-Verlag, Wiesbaden.

Oliver, I. M., Smith, D. J. and Holland, J. R. C. (1987) A study of permutation crossover operators on the traveling salesman problem. In *Genetic Algorithms and their Applications*, ed. J. J. Grefenstette, pp. 224-225. Lawrence Erlbaum Associates, Hillsdale/NJ.

Or, I. (1976) Traveling salesman-type combinatorial problems and their relation to the logistics of blood banking. Ph.D. thesis, Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL.

Osman, I. H. (1993) Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, **41**, 421-451.

Osman, I. H. (1995) An introduction to meta-heuristics. In *Operational Research Tutorial Papers*, ed. M. Lawerence and C. Wilson, pp. 92-122. Operational Research Society Press, Birmingham.

Potvin, J.-Y. and Rousseau, J. M. (1995) An exchange heuristic for routing problems with time windows. *Journal of the Operational Research Society*, **46**, 1433-1446.

Potvin, J.-Y., Kervahut, T., Garcia, B.-L. and Rousseau, J.-M. (1996) The vehicle routing problem with time windows - part I: tabu search. *INFORMS Journal on Computing*, **8**, 158-164.

Potvin, J.-Y. and Bengio, S. (1996) The vehicle routing problem with time windows - part II: genetic search. *INFORMS Journal on Computing*, **8**, 165-172.

Rechenberg, I. (1973) *Evolutionsstrategie*. Fromman-Holzboog, Stuttgart.

Retzko, R. (1995) *Flexible Tourenplanung mit selbstorganisierenden Neuronalen Netzen*. Unitext, Göttingen.

Rochat, Y. and Taillard, E. D. (1995) Probabilistic diversification and intensification in local search for vehicle routing. Technical report CRT-95-13, Centre de recherche sur les transports, Université de Montréal, Montréal.

Russell, R. A. (1995) Hybrid heuristics for the vehicle routing problem with time windows. *Transportation Science*, **29**, 156-166.

Schwefel, H.-P. (1977) *Numerische Optimierung von Computer-Modellen mittels der Evolutions-strategie*. Birkhäuser, Basel.

Schwefel, H.-P. (1987) Collective phenomena in evolutionary systems. Interne Berichte und Skripten Nr. 233, Fachbereich Informatik, Universität Dortmund, Dortmund.

Solomon, M. M. (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, **35**, 254-265.

Solomon, M. M., Baker, E. K. and Schaffer, J. R. (1988) Vehicle routing and scheduling problems with time window constraints: efficient implementations of solution improvement procedures. In *Vehicle Routing: Methods and Studies*, ed. B. L. Golden and A. A. Assad, pp. 85-105. North-Holland, Amsterdam.

Taillard, E., Badeau, P., Gendreau, M., Guertin, F. and Potvin, J.-Y. (1996) A tabu search heuristic for the vehicle routing problem with soft time windows. Technical report CRT-95-66, Centre de recherche sur les transports, Université de Montréal, Montréal.

Thangiah, S. R., Nygard, K. E. and Juell, P. L. (1991) GIDEON: a genetic algorithm system for vehicle routing with time windows. In *Proceedings of the 7th Conference on Artificial Intelligence for Applications*, pp. 322-328. IEEE Press, Miami, FL.

Thangiah, S. R., Osman, I. H. and Sun, T. (1994) Hybrid genetic algorithms, simulated annealing and tabu search methods for vehicle routing problems with time windows. Technical Report UKC/OR94/4, Institute of Mathematics & Statistics, University of Kent, Canterbury, UK.

Parameter	Evolution strategies	
	ES1	ES2
λ	50	450
μ	8	45
κ	4	20

Table 1. Values of the parameters λ, μ und κ.

Problem Class	Values	Rochat and Taillard 1995	Russell 1995	Chiang and Russell 1996	Potvin et al. 1996	Potvin and Bengio 1996	Taillard et al. 1996	Bachem et al. 1997	Chiang and Russell 1997	Liu and Shen 1998	ES1	ES2
R1	MNV	12.58	12.66	12.50	12.60	12.60	12.25	12.25	12.17	12.17	**11.92**	12.00
	MND	**1197.42**	1317.00	1308.82	1294.70	1296.83	1216.70	1264.24	1204.19	1249.57	1228.06	1226.38
	MCT	2700	81	206	639	679	13774	611	5395	2657	750	1176
R2	MNV	3.09	2.91	2.91	3.10	3.00	3.00	2.91	**2.73**	2.82	**2.73**	**2.73**
	MND	**954.36**	1167.00	1166.42	1185.90	1117.70	995.38	1100.33	986.32	1016.58	969.95	1033.58
	MCT	9800	116	273	722	2384	20232	674	6115	398	960	1302
C1	MNV	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**
	MND	828.45	930.00	909.80	861.00	838.00	828.45	829.50	**828.38**	830.06	**828.38**	**828.38**
	MCT	3200	71	139	435	601	14630	-	644	1320.60	522	570
C2	MNV	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
	MND	590.32	681.00	684.10	602.50	589.90	590.30	591.88	591.42	591.03	**589.86**	**589.86**
	MCT	7200	76	166	431	2482	16375	-	1440	215	780	1062
RC1	MNV	12.38	12.38	12.38	12.60	12.10	11.88	11.75	11.88	11.88	11.63	**11.50**
	MND	**1369.48**	1523.00	1473.90	1465.00	1446.20	1367.51	1414.63	1397.44	1412.87	1392.57	1406.58
	MCT	2600	67	177	586	673	11264	570	2840	1828	660	1407
RC2	MNV	3.62	3.38	3.38	3.40	3.40	3.38	3.38	**3.25**	**3.25**	**3.25**	**3.25**
	MND	1139.79	1398.00	1401.50	1476.10	1360.60	1165.62	1341.35	1229.54	1204.87	1144.43	1175.98
	MCT	7800	103	238	662	2134	11596	499	3866	427	990	1280
All	CNV	427	424	422	427	422	416	414	411	412	**406**	**406**
	CND	**57120**	65827	65201	64679	62572	57993	61523	58502	59317	57876	58921

Table 2. Comparison of results for the problem instances from Solomon (1987).

Problem instance	Best published solution			Best computed solution ES1		ES2	
	NV	ND	Reference	NV	ND	NV	ND
R101	**18**	**1607.70**	Desrochers et al. (1992)	19	1650.80	19	1656.79
R102	**17**	**1434.00**	Desrochers et al. (1992)	17	1486.12	17	1490.39
R103	**13**	**1207.00**	Thangiah et al. (1994)	13	1292.85	13	1292.88
R104	10	982.01	Rochat and Taillard (1995)	**9**	**1013.32**	9	1030.78
R105	**14**	**1377.11**	Rochat and Taillard (1995)	14	1378.88	**14**	**1377.11**
R106	**12**	**1252.03**	Rochat and Taillard (1995)	12	1272.83	12	1260.12
R107	10	1124.72	Chiang and Russell (1997)	10	1139.00	**10**	**1120.85**
R108	**9**	**968.59**	Taillard et al. (1996)	9	970.05	9	973.69
R109	11	1214.54	Taillard et al. (1996)	**11**	**1194.73**	12	1217,02
R110	**10**	**1174.49**	Chiang and Russell (1997)	10	1182.49	10	1188.44
R111	10	1104.83	Taillard et al. (1996)	10	1160.90	**10**	**1099.46**
R112	10	953.63	Rochat and Taillard (1995)	**9**	**1003.73**	9	1009.04

Problem instance	Best published solution			Best computed solution ES1		ES2	
	NV	ND	Reference	NV	ND	NV	ND
R201	4	1254.80	Taillard et al. (1996)	4	1255.76	**4**	**1252.37**
R202	3	1214.28	Taillard et al. (1996)	3	1199.80	**3**	**1198.45**
R203	3	948.74	Rochat and Taillard (1995)	**3**	**942.64**	3	1139.11
R204	2	855.21	Chiang and Russell (1997)	**2**	**854.88**	2	940.77
R205	3	1035.60	Chiang and Russell (1997)	**3**	**1013.47**	3	1162.06
R206	**3**	**833.00**	Thangiah et al. (1994)	3	913.68	3	973.47
R207	**2**	**914.39**	Chiang and Russell (1997)	2	980.57	2	971.34
R208	2	738.60	Rochat and Taillard (1995)	2	732.61	**2**	**731.23**
R209	**3**	**855.00**	Thangiah et al. (1994)	3	910.55	3	982.12
R210	3	967.50	Rochat and Taillard (1995)	**3**	**955.39**	3	998.38
R211	2	923.80	Taillard et al. (1996)	**2**	**910.09**	2	1020.08

Problem instance	Best published solution			Best computed solution ES1		ES2	
	NV	ND	Reference	NV	ND	NV	ND
C101	10	827.30	Desrochers et al. (1992)	10	828.94	10	828.94
C102	10	827.30	Desrochers et al. (1992)	10	828.94	10	828.94
C103	10	828.06	Rochat and Taillard (1995)	10	828.06	10	828.06
C104	10	824.78	Rochat and Taillard (1995)	10	824.78	10	824.78
C105	10	828.94	Potvin and Bengio (1993)	10	828.94	10	828.94
C106	10	827.30	Desrochers et al. (1992)	10	828.94	10	828.94
C107	10	827.30	Desrochers et al. (1992)	10	828.94	10	828.94
C108	10	827.30	Desrochers et al. (1992)	10	828.94	10	828.94
C109	10	828.94	Potvin and Bengio (1993)	10	828.94	10	828.94

Problem instance	Best published solution			Best computed solution ES1		ES2	
	NV	ND	Reference	NV	ND	NV	ND

C201	3	591.56	Potvin and Bengio (1993)	3	591.56	3	591.56
C202	3	591.56	Potvin and Bengio (1993)	3	591.56	3	591.56
C203	3	591.17	Rochat and Taillard (1995)	3	591.17	3	591.17
C204	3	590.60	Potvin and Bengio (1993)	3	590.60	3	590.60
C205	3	588.88	Potvin and Bengio (1993)	3	588.88	3	588.88
C206	3	588.49	Potvin and Bengio (1993)	3	588.49	3	588.49
C207	3	588.29	Rochat and Taillard (1995)	3	588.29	3	588.29
C208	3	588.32	Rochat and Taillard (1995)	3	588.32	3	588.32

Problem instance	Best published solution			Best computed solution ES1		ES2	
	NV	ND	Reference	NV	ND	NV	ND
RC101	**14**	**1669.00**	Thangiah et al. (1994)	14	1701.06	14	1697.43
RC102	**12**	**1554.75**	Taillard et al. (1996)	12	1571.89	12	1558.07
RC103	**11**	**1110.00**	Thangiah et al. (1994)	11	1263.09	11	1272.51
RC104	**10**	**1135.83**	Rochat and Taillard (1995)	10	1143.25	10	1138,57
RC105	13	1643.38	Taillard et al. (1996)	13	1693.70	**13**	**1637.15**
RC106	11	1448.26	Taillard et al. (1996)	12	1384.11	**11**	**1432.12**
RC107	**11**	**1230.54**	Taillard et al. (1996)	11	1232.26	11	1369.52
RC108	**10**	**1139.82**	Taillard et al. (1996)	10	1151.22	10	1147.26

Problem instance	Best published solution			Best computed solution ES1		ES2	
	NV	ND	Reference	NV	ND	NV	ND
RC201	**4**	**1249.00**	Thangiah et al. (1994)	4	1415.00	4	1418.86
RC202	3	1445.86	Liu and Shen (1998)	**3**	**1389.57**	3	1665.56
RC203	3	1078.73	Chiang and Russell (1997)	**3**	**1060.45**	3	1065.02
RC204	3	806.75	Rochat and Taillard (1995)	3	806.48	**3**	**799.12**
RC205	4	1322.81	Liu and Shen (1998)	4	1321.01	**4**	**1302.42**
RC206	**3**	**1158.81**	Taillard et al. (1996)	3	1196.12	3	1196.12
RC207	**3**	**1082.32**	Taillard et al. (1996)	3	1116.78	3	1112.60
RC208	**3**	**833.97**	Rochat and Taillard (1995)	3	850.02	3	848.10

Table 3. Best results for the problem instances from Solomon (1987).

Number of customers	Tours
R104	
13	6-94-96-59-93-99-84-17-45-83-5-60-89
12	12-80-68-24-29-79-78-34-81-33-77-3
10	18-82-48-46-8-47-36-49-19-7
10	21-75-56-23-67-39-55-4-25-54
11	28-1-69-76-53-26-40-13-95-97-58
11	42-43-15-57-87-2-73-72-74-22-41
11	52-88-62-11-64-63-90-32-10-31-27
10	70-30-20-66-9-35-65-71-51-50
12	92-98-14-44-38-86-16-61-85-91-100-37
	Number of vehicles: 9 Total travel distance: 1013.32
R112	
11	2-41-22-72-74-75-56-23-67-39-4
11	5-61-16-86-38-14-44-91-100-37-96
11	21-73-40-53-26-12-80-24-25-55-54
11	27-31-88-7-82-8-46-36-49-47-48
11	28-76-79-78-34-35-71-65-66-20-1
10	52-62-19-11-64-63-90-32-10-70
11	69-30-51-9-81-33-50-29-3-77-68
12	92-98-85-93-87-57-15-43-42-97-13-58
12	95-59-99-94-6-18-83-84-17-45-60-89
	Number of vehicles: 9 Total travel distance: 1003.73
RC202	
33	65-82-12-14-47-15-11-83-64-19-23-21-48-18-76-51-22-86-87-9-57-52-10-97-59-74-13-17-7-4-60-100-70
35	91-92-95-85-63-33-28-26-27-29-31-30-62-67-71-61-41-38-40-81-90-84-49-20-66-56-50-34-32-89-24-25-77-75-58
32	98-45-5-3-1-42-36-39-44-69-88-73-16-99-53-78-79-8-6-46-2-55-68-54-43-35-37-72-96-93-94-80
	Number of vehicles: 3 Total travel distance: 1389.57

Table 4. New best-known solutions.

Reference	Problem instance D-417		Problem instance E-417	
	NV	ND	NV	ND
Thangiah et al. (1991)	54	4866	55	4149
Kontoravdis and Bard (1995)	55	4273	55	4986
Rochat and Taillard (1995)	54	6265	54	7212
Russell (1995)	55	4964	55	6092
Chiang and Russell (1996)	55	4232	55	4397
Taillard et al. (1996)	55	3440	55	3707
Chiang and Russell (1997)	55	3455	55	3797
Liu and Shen (1998)	**54**	**3747**	**54**	**4569**
Homberger and Gehring ES1	54	4703	54	4732
Homberger and Gehring ES2	54	9708	55	5174

Table 5. Comparison of results for two problem instances from Russell (1995).

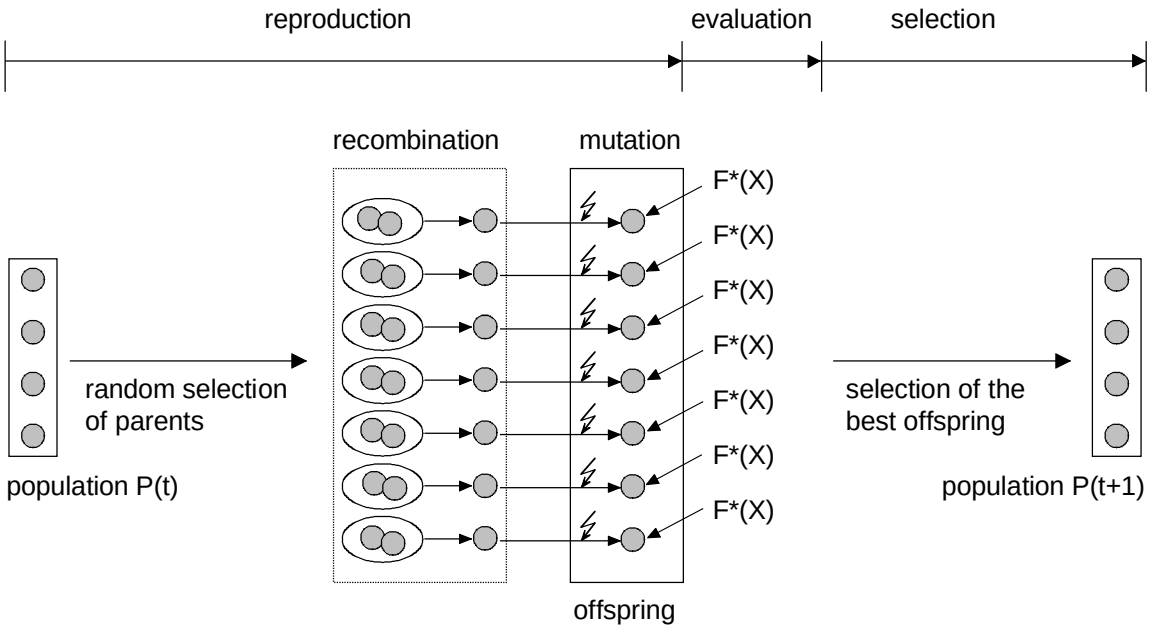


Figure 1. (μ, λ)-evolution strategy.

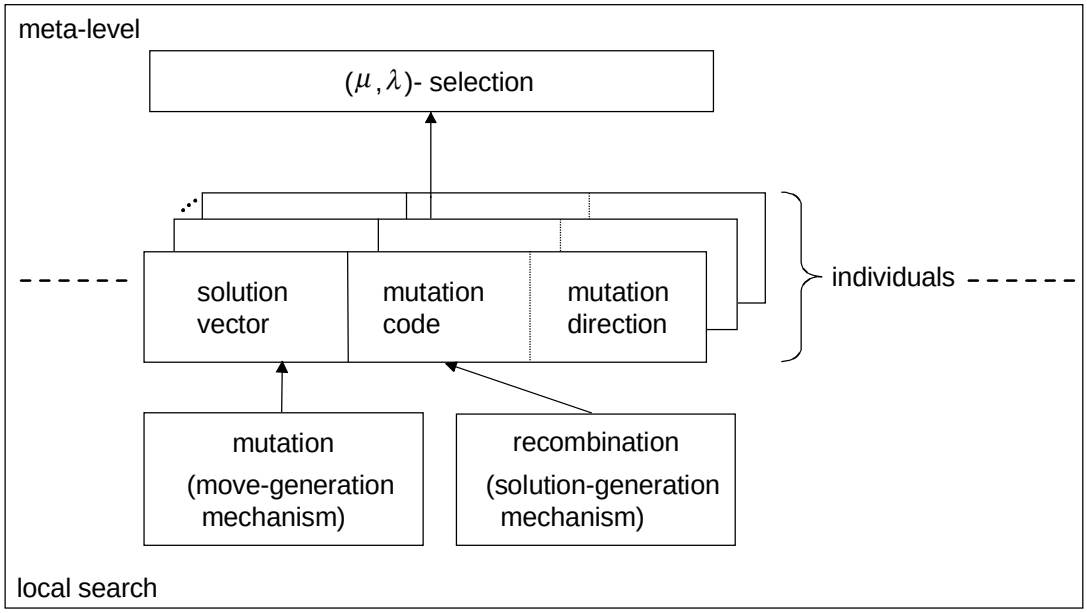


Figure 2. Concept of an evolutionary metaheuristic.

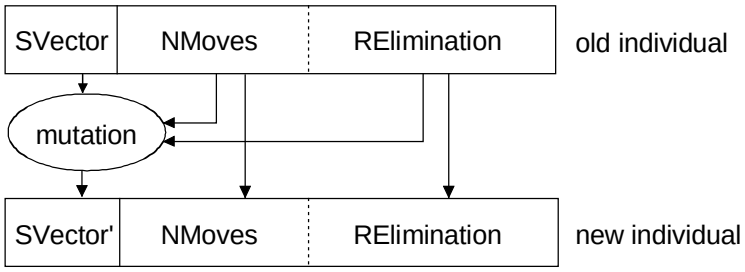
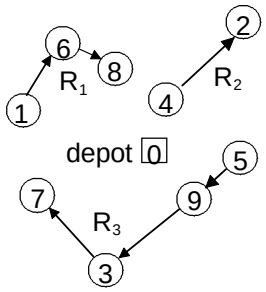


Figure 3. Generation of a new individual in the case of evolution strategy ES1.



a) Representation as a graph

SVector = (0,1,6,8,0,0,4,2,0,0,5,9,3,7,0)
where:

R_1 : (0,1,6,8,0)
R_2 : (0,4,2,0)
R_3 : (0,5,9,3,7,0)

b) Representation as a solution vector

Figure 4. Representation of a route schedule.

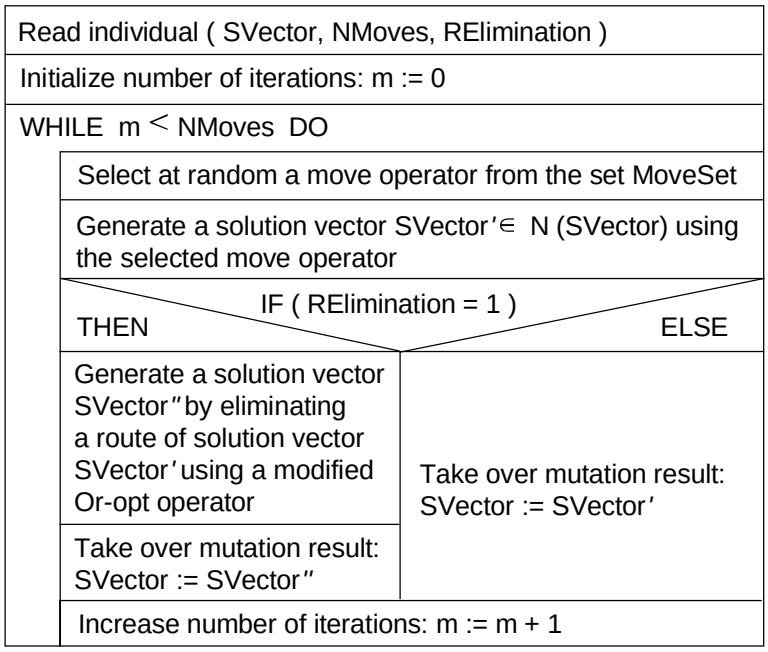


Figure 5. Mutation rule of evolution strategy ES1.

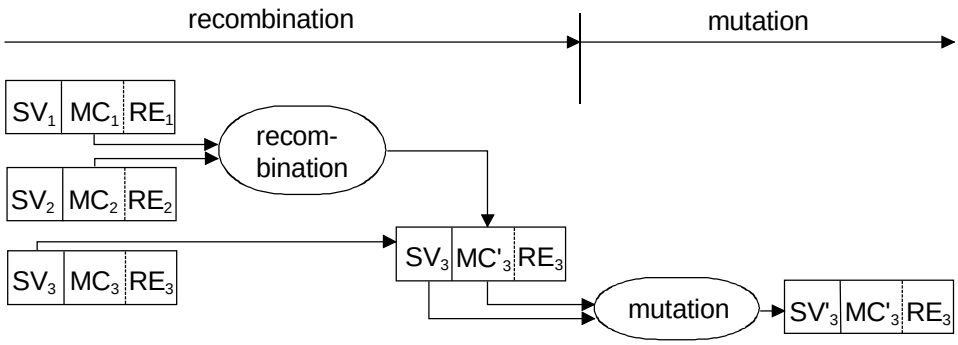


Figure 6. Generation of a new individual in the case of evolution strategy ES2.

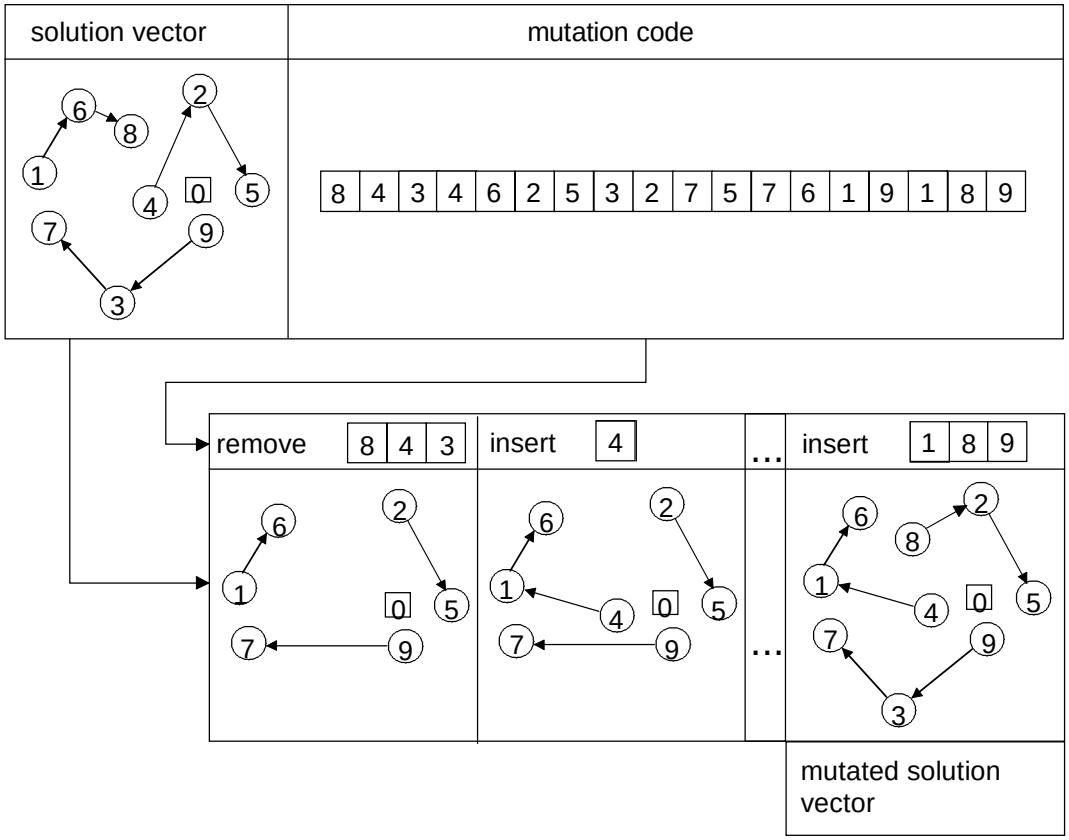


Figure 7. Simplified example of the mutation of an individual.

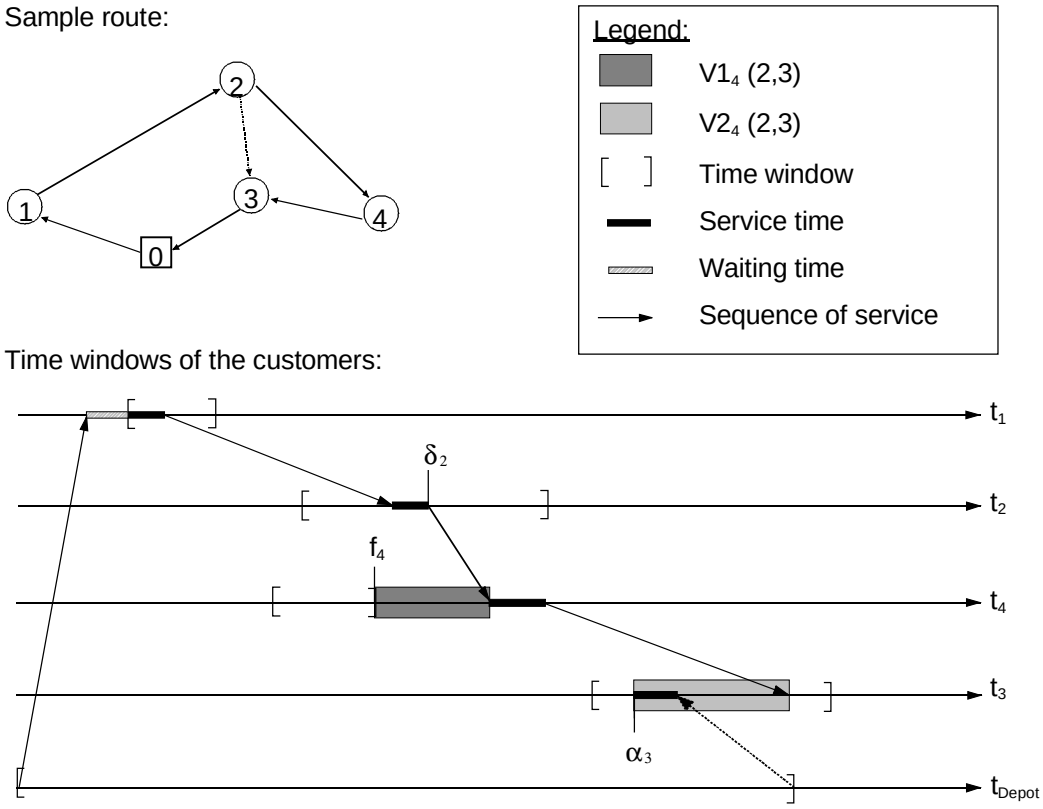


Figure 8. Computation of the time shifts $V1_k(i, i+)$ and $V2_k(i, i+)$.