APPLICATION OF GENETIC ALGORITHMS TO VEHICLE ROUTING PROBLEM

Denisa Mocková, Alena Rybičková[*]

Abstract: Distribution of the goods from a producer to a customer is one of the most important tasks of transportation. This paper focuses on the usage of genetic algorithms (GA) for optimizing problems in transportation, namely vehicle routing problem (VRP). VRP falls in the field of NP-hard problems, which cannot be solved in polynomial time. The problem was solved using genetic algorithm with two types of crossover, both including and leaving-out elitism, setting variable parameters of crossover and mutation probability, as well as prevention of creating invalid individuals. The algorithm was programmed in Matlab, tested on real world problem of spare parts distribution for garages, while the results were compared with another heuristic method (Clarke-Wright method). Genetic algorithm provided a better solution than the heuristic Clarke-Wright method.

Key words: *VRP problem, genetic algorithms, Clarke-Wright algorithm, metaheuristics*

Received: November 2, 2012
Revised and accepted: January 8, 2014

DOI: 10.14311/NNW.2014.24.003

1. Introduction

Vehicle routing problem can be described as a problem of determining vehicle routes, where every route starts in the depot, then a subset of customers is visited in a specific sequence, followed by return to the depot. Every customer has to be assigned to exactly one route and the overall amount of delivered quantity of consignments assigned to one vehicle cannot exceed the capacity of the vehicle. Routes should be chosen so that the total cost of delivery is the minimum.

VRP can be considered as a generalization of the travelling salesman problem (TSP). Unlike TSP, vehicle routing problem has a wide range of other limitations and extensions that can be often seen in real-world applications. These can include for example:

– distribution is ensured from multiple depots;

– every vehicle can operate on more than one route provided that the total time does not exceed the given value;

[*] Denisa Mocková – Corresponding Author, Alena Rybičková, Czech Technical University in Prague, Faculty of Transportation Sciences, Department of Logistic and Transportation Processes, Horská 3, `mockova@fd.cvut.cz, alena.rybickova@gmail.com`

- every customer has to be served in given time period which is called time window;
- the problem can include both delivery to and pick-up from the customers;
- use of homogeneous or heterogeneous fleet, where one or multiple type of vehicles is used;
- vehicles can be set with a given time period of operation, etc.

Variants of VRP have been extensively studied in literature (for detailed review see [1, 2, 3, 4]).

2. VRP solution methods

Exact methods

Small scale problems can be solved using exact approaches which subsequently browse and evaluate all possible solutions of the problem. Well-known examples are branch-and-bound and branch-and-cut methods. Application of branch and cut algotithm for VRP can be found in [5].

Heuristics

Classic heuristics are methods that do not ensure the finding of an optimal solution, but ensure that a good enough solution is found in a reasonable time. An example of a heuristic method for VRP is the Clarke-Wright algorithm [6].

Metaheuristics

Metaheuristics are methods intended for solving very general types of problems. We can understand them as a general frame which can be applied to a wide range of different optimization problems with relatively few modifications required for adjusting the algorithm to the specific problem [7].

Unlike classic heuristics, in metaheuristics it is possible to abandon the local optimum in cases where different area of feasible solutions offers a hope of finding a solution with a better objective function value. For this purpose metaheuristics usually use a compromise between randomization and local search. In literature the terms heuristics and metaheuristics are often confused, but the development leads toward the use of the term metaheuristics for methods that include randomness.

Various metaheuristics were applied to VRP, mainly to some specific variants, such as VRP with time windows. Regarding basic capacitated vehicle routing problem ant colony optimization [8], simulated annealing [9], genetic algorithm [10], tabu search [11] were used.

In this paper we will show a modified genetic algorithm for a VRP.

2.1 Clarke-Wright method

Clarke-Wright algorithm works on the principle of cost savings gained by joining two routes into one. Principle of savings calculation is depicted in Fig. 1.

Initially customers i and j are visited on separate routes (1a). An alternative to this is to visit the two customers on the same route (1b). Because the transportation costs are given, the savings S_{ij} that result from using one route instead of two routes can be calculated as follows:

$$S_{ij} = D_a - D_b = c_{i0} + c_{0j} - c_{ij}$$

where c_{ij} represent transportation costs between two given points and D_a (D_b) total transportation costs in Fig. 1a (b).

In the first step of the savings algorithm the savings for all pairs of customers are calculated, and all pairs of customer points are sorted in descending order of the savings. Subsequently, from the top of the sorted list of point pairs one pair is considered at a time. When a pair of points $i - j$ is considered, the two routes that visit i and j are combined (such that j is visited immediately after i on the resulting route), if this can be done without deleting a previously established direct connection between two customer points, and if the total demand on the resulting route does not exceed the vehicle capacity. After all pairs of points are considered, the algorithm returns a list of routes.

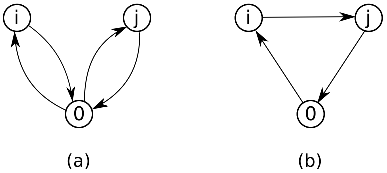


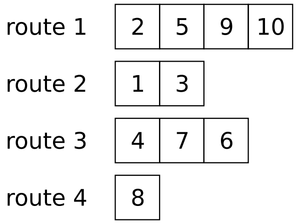
Fig. 1 *Clarke-Wright algorithm savings principle*

2.2 Genetic algorithm

Genetic algorithms belong together with evolution strategies and evolution programming to main areas of evolution algorithms. Genetic algorithms can be distinguished from later two by representation of the individuals and recombination operator. Genetic algorithms emphasize genetic coding of potential solutions into chromosomes and application of genetic operators (crossover) to these chromosomes. Genetic representation is essential for the success of the algorithm, a well-chosen representation will facilitate the solution of the problem, while a bad representation will have the opposite effect.

Representation

Genome representation was chosen in such a way as to most resemble the natural representation of the problem. Every genome (individual) is constituted of a set of point lists, while the points are listed in the same order in which they are visited on the route. Natural numbers are used for the representation. An example of the representation of one genome with ten visited points is given in Fig. 2.



route 1	2	5	9	10
route 2	1	3		
route 3	4	7	6	
route 4	8			

Fig. 2 *Example of genome representation with ten visited vertexes*

During the initialization the individuals are created randomly but all individuals have to comply with the given limitations, therefore in each step of the algorithm there is a check process in place which ensures that all individuals are feasible. If there is an individual with a length of route or a number of vertexes on one route exceeding the given values, the route is divided into several new ones that meet the limitations.

Fitness function

The choice of the fitness function is given by the assignment of the problem, which is to minimize the total distance travelled. Therefore,the fitness value is computed as a simple sum of lengths of individual routes and the criterion is the minimization.

Selection

As a selection method the tournaments selection was used with 3 individuals randomly chosen in each step,out of which the best one regarding the fitness value is selected. This individual then enters the crossover process.

Crossover

Because of the representation system we cannot use basic crossover methods such as one or multiple point crossover when parental solutions are divided in one or several points and then joined together. If such a crossover were to be used, solutions with duplications and therefore not valid solutions would appear. Instead we used a different crossover operator, where a random section of the route is chosen in the first parental solution and this section is inserted in the second parental solution. Afterwards all values from inserted section are deleted from the second parent so that all values appear only once in the newly created individual.

Example of this crossover is displayed in Fig. 3.

The position where the chosen section is inserted can be chosen randomly. A different option which is described in paper [12] is to find a point that is nearest to the first point of the inserted section (and does not occur in the inserted section) and insert the section directly behind this nearest point.

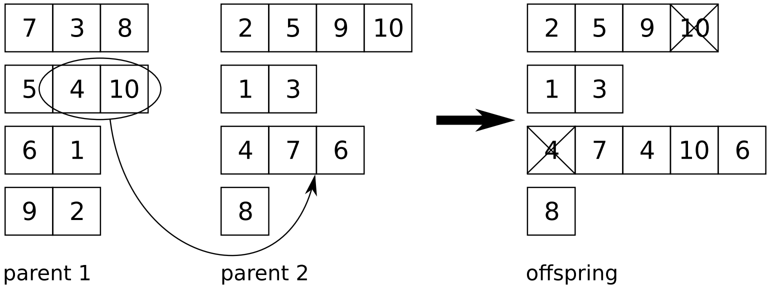


Fig. 3 *Example of the crossover*

During the testing we used both variants, as described below. We also tested a variant with randomly altering both types of crossover. The probability assigned to both options was equal, i.e. 0.5.

The method with the nearest point insert ensures faster convergence of the solution. On the other hand though, it could also cause that a large area of possible solutions would not be visited.

Mutation

Likewise, as in the case of crossover operator, the mutation has to be adjusted to the problems characteristics. Mutation is based on altering two randomly chosen points in the solution. A different option that could be used in this case is to choose a random point and insert it in the different position of the solution.

Elitism

Both variants – including and leaving out elitism – were tested. In the case of including elitism, one individual with the best fitness function value is copied into the next generation.

Mutation and crossover parameters

Crossover and mutation operators are not applied for every offspring, they are used with a given probability. First, two parents are chosen using a selection operator, then random number from the interval (0,1) is generated and if this number is lower than the crossover probability, the parents enter the crossover process. If the generated number is higher than the probability value, the first parent is directly copied into the next generation and the second parent is not used. The mutation works by analogy – the mutation operator is applied only if the generated number is smaller than the given mutation probability.

Based on the recommended values of crossover and mutation probability [13], the program was executed for all possible combinations of the following parameter values:

- crossover probability: 0.70; 0.80; 0.85; 0.90; 0.95;
- mutation probability: 0.05; 0.10; 0.15.

3. Solution of the VRP

3.1 Description of the problem

The application of both methods (Clarke-Wright and genetic algorithm) is shown on the problem of the optimization of spare parts distribution of two big car manufacturers in the Czech Republic [14, 15].

The market in spare parts is a stable component of the car industry, which has been developing over the long term and recently considerably gaining in its importance. One of the key elements in the network of authorised garages is spare parts delivery, which has to be flexible, fast and accurate. The distribution comprises complex operations with a multilevel supply to roughly hundreds to thousands of destinations from various manufacturers, while there is a requirement for deep knowledge of the issue as well as for speed and minimum error rate. Demand for spare parts is mostly of a random nature and therefore it is difficult to define the stock size. However, for the garages it is not convenient to hold a high stock of the spare parts, and for this reason the supply is solved in the form of orders for the required goods stocked at the central depot and their flexible delivery.

The key contributor to the growth of this market is mainly the growth in the average car age and the related necessity for a larger number of repairs. Furthermore there is a change in the repair process, where the damaged parts are replaced by the new ones instead of their being repaired and, last but not least,the growth in the spare parts price. On the other hand the reliability of the cars increases, which leads to longer intervals between garage visits.

The distribution logistics thus opens a significant space for the application of software products enabling effective management and cost minimization of transport processes that require quick adaptation to fast-changing conditions.

Current distribution system

Each working day the authorised garages order spare parts at the central depot in Austria by 16:00. There are two modes of such orders:

- *Urgent* – spare parts ordered must be prepared and sent to the Czech Republic on the same day. At night the trucks arrive at the depots in Brno and Prague (garages are divided by region) and by 8:00 of the following day the spare parts must be delivered to the garages.

- *Stock* – applies to the spare parts intended for more common repairs and therefore for the predictable demand. In this mode spare parts have to be delivered to the garages within 48 hours from the order time.

From the Prague and Brno depots the spare parts are distributed and unloaded in the night depots, i.e. rooms or separate buildings of the garages to which the drivers have the keys, entry codes, etc. The consignments are not physically taken over by the recipient; the recipient checks them only after the beginning of working hours. These persons confirm the delivery notes and leave them at the night depots where from the drivers collect them at the next delivery.

Spare parts distribution from both depots is provided every day from Monday to Friday on 10 routes from the Prague depot and 5 routes from the Brno depot. Distribution of the spare parts in Urgent and Stock modes is made jointly by one vehicle. For determining the route lengths and travel times of the routes the Internet application Google Maps was used. Travel times do not include stop-overs at the garages. Where multiple variants were available, the option with the shortest travel time was chosen. Both depots supply 97 points in the Czech Republic, of which 66 points are supplied from the Prague depot and 31 from the Brno depot.

Our model of this distribution system used homogenous fleet with limiting conditions of total distance travelled on one route and number of garages visited on one route. The maximum number of garages visited corresponds to the limited capacity of the vehicle. If the weight and the amount of distributed parts is known, the maximum loading weight could be chosen as the limiting condition. However, the load weight changes every day, while the route optimisation is supposed to be suggested for a longer period. Therefore the weights were replaced by the maximum number of garages on the route that corresponds to the average quantities of distributed spare parts. Prague and Brno depot were solved separately.

3.2 Clarke-Wright algorithm

For various combinations of limiting conditions several possible solutions were chosen. Values of the limiting conditions were chosen so as to roughly match the current solution. The best results were obtained with the initial conditions of 400 km as the maximum route length and the maximum of 8 garages on the route.

Given the limiting conditions of 400 km and 8 points on one route, we got 8 distribution routes from the Prague depot and 5 routes from the Brno depot. The total minimum length of all routes from both depots was found to be 1578 + 2492 km, i.e. in total 4070 km (I). These values will serve for basic comparison of the success rate of the suggested metaheuristic algorithms.

Routes found using CW algorithm are displayed in the map of the Czech Republic in the Fig. 4.

	Prague current	**Prague CW**	**Brno current**	**Brno CW**
max. number of garages on the route	8	8	8	8
max. route length [km]	388	400	475	400
number of routes	10	8	5	5
total length [km]	2942.9	2492.1	1812.0	1577.6
length save [%]	–	15.32	–	12.94

Tab. I *Overview of the solutions obtained from Clarke-Wright algorithm and their comparison with the current distribution system.*

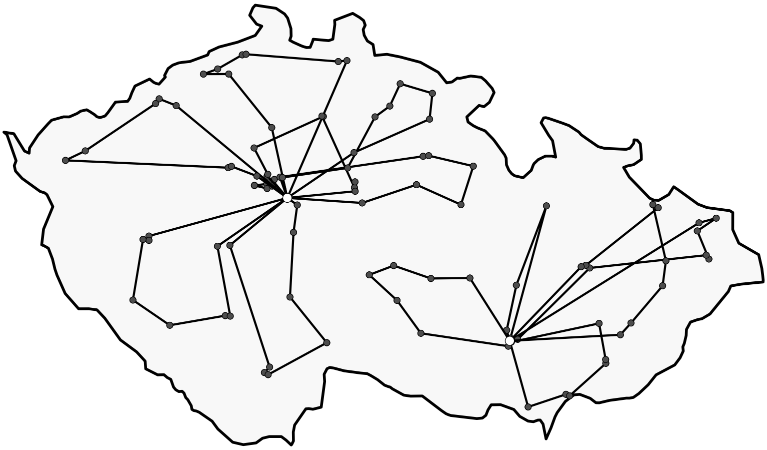


Fig. 4 *Routes found using Clarke-Wright algorithm.*

3.3 Genetic algorithm

The genetic algorithm will be used on the same data, while keeping the depots positions unchanged. The specific implementation scheme results from [12]. The most important issue in the case of genetic algorithms used for VRP is the representation of individual solutions and also making sure that no duplicates are created during the crossover and mutation.

The behaviour of the genetic algorithm was tested with the following parameter settings:

– crossover type (random insert crossover, nearest point insert crossover, combination of both types),

– crossover probability (0.70, 0.80, 0.85, 0.9, 0.95),

– mutation probability (0.05, 0.10, 0.15),

– population size (20, 30, 60),

– including elitism (yes, no).

The number of iterations was limited to 20000 when the best value found changes very slightly.

Individual variants and parameter setting influence will be first compared on a part of the problem which includes the garages falling under to the Prague depot. The variant giving the best results will be then used for proposing routes from the Brno depot.

Implementation

For the algorithms implementation, Matlab environment was chosen. Regarding the different route lengths in individual solutions, cell arrays were used for saving the individuals. Cell arrays enable saving different data types in each cell, in our case number vectors of different lengths. This makes it possible to reduce the memorys requirements in comparison with the usage of fixed size array.

On the other hand the computation time is greater than in other programming languages. Considering algorithm execution with tens of thousands population sizes these differences become significant. If the solution is required in a shorter time, another programming language should be chosen, such as language C.

Depot Prague

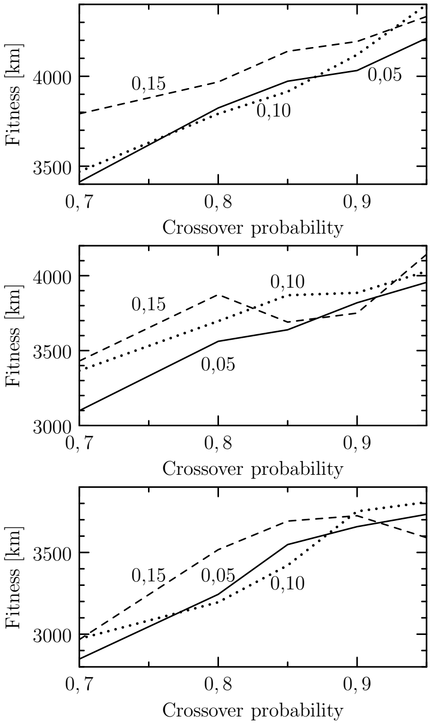
Variant 1 – random insert crossover without elitism

The first variant tested is the genetic algorithm using crossover with random section transfers between parents. The minimum fitness function values for all combinations of crossover and mutation probability and population size are summed up in Tab. II, including their graphs.

Population size 20		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	3412.2	3469.6	3791.7
	0.80	3823.9	3791.6	3968.5
	0.85	3972.7	3914.7	4139.1
	0.90	4032.2	4119.9	4193.6
	0.95	4211.9	4398.5	4332.4

Population size 30		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	3099.0	3366.1	3430.6
	0.80	3561.9	3696.5	3873.8
	0.85	3638.8	3869.4	3690.2
	0.90	3819.2	3885.7	3750.1
	0.95	3955.3	4029	4141.9

Population 60		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2847.9	2971.5	2967.1
	0.80	3243.7	3195.3	3516.9
	0.85	3548.2	3421	3691.9
	0.90	3657.8	3751.6	3724.6
	0.95	3733.1	3807.4	3590.5



Tab. II *Minimum fitness function values for variant 1 – in the graphs fitness function values are displayed based on mutation and crossover probability. All values were obtained after 20 000 iterations. Tables and graphs are displayed separately for population sizes of 20, 30 and 60 individuals. Mutation probability is constant for each displayed curve.*

If we compare the obtained values and the Clarke-Wright algorithm results (2492.1 km), it is obvious that the genetic algorithm results are far from an optimal solution (which has to be smaller or equal to CW results). Even in the case of the best value found of 2847.9 km as the total length of all routes, the difference exceeds 14% against the CW.

The graphs and Tab. II show significant dependence on the parameter settings. Mutation probability has only a small impact; on the contrary changes in the crossover probability imply bigger differences in the results. Against expectations, better results were obtained with lower crossover probability values (0.7), although this value is lower than the generally recommended values (according to [2] around 0.9). It is probable that even small changes caused by crossover or mutation can significantly influence the length of the routes and result in a loss of good solutions. With low crossover probability the good solutions are copied to the next generation without any change and therefore final fitness values are better. For this reason elitism was included in the variants 4, 5 and 6.

The population size dependence can be seen in the figures when for the greater population sizes the results obtained are better, which corresponds to the fact that more solutions are created and tested.

One program run progress is displayed in Fig. 5. After several hundred steps almost no convergence of the maximum, minimum and average fitness value can be

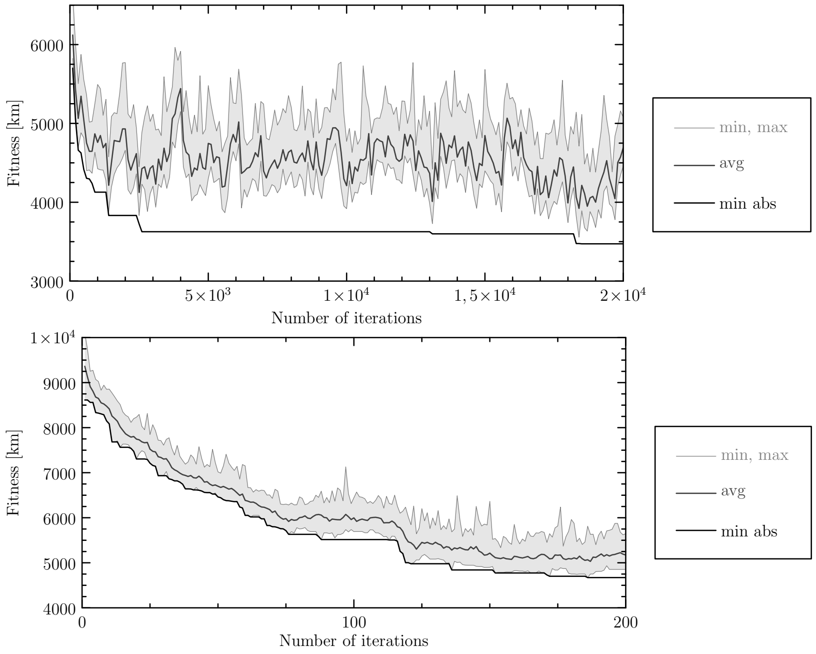


Fig. 5 *Algorithm progress – variant 1. The graph depicts the behaviour of one algorithm run with the following initial values: 30 individuals in the population, mutation probability 0.1, and crossover probability 0.8. In the top graph the maximum fitness value of the population (max), average fitness value of the population (avg), minimum fitness value of the population (min), and absolute minimum since the beginning of the program run (min abs) are displayed after each 100 of 20 000 steps. The bottom graph displays the same values for the first 200 steps.*

observed. On the other hand the average and minimum fitness value in individual populations often diverges noticeably from the earlier found absolute minimum fitness value. In the Fig. 5 (bottom) the first two hundred steps of the algorithm for the same initial values are displayed.

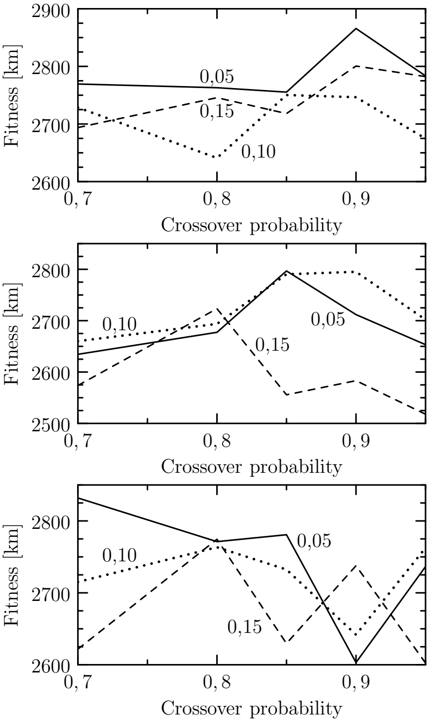
Variant 2 – nearest point crossover without elitism

The second tested variant uses crossover in which section from parent 1 is randomly chosen and inserted in parent 2 at such a point that is the nearest point to the first point of inserted section. The minimum fitness function values found are summarized in Tab. III.

Population size 20		Mutation probability		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2769.4	2728.6	2693.8
	0.80	2763.0	2641.0	2745.7
	0.85	2755.3	2754.3	2718.1
	0.90	2865.8	2746.5	2800.6
	0.95	2783.5	2674.7	2782.3

Population size 30		Mutation probability		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2634.5	2660.1	2573.8
	0.80	2677.3	2693.7	2723.3
	0.85	2796.8	2794.3	2555.7
	0.90	2711.8	2795.1	2583.2
	0.95	2653.3	2702.0	2517.9

Population size 60		Mutation probability		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2831.9	2715.1	2621.8
	0.80	2771.2	2763.4	2774.5
	0.85	2780.8	2732.4	2629.6
	0.90	2603.1	2641.8	2737.7
	0.95	2736.5	2762.6	2602.4



Tab. III *Minimum fitness function values for variant 2 – for legend description see Tab. II.*

The change of the crossover type influenced the results notably and the minimum fitness value oscillates between 2517.9 and 2865.8 for various parameter values. The results do not show any significant dependence on the parameter settings and a relatively good solution can be obtained with any initial setting of probabilities and population size. Comparing the best value found and the CW results, there is only a small difference, although the solution with a shorter total length failed to be found. The difference in this case is approximately 1%.

Fig. 6 shows the behaviour of the algorithm over time. Unlike random crossover, the population minimum and absolute minimum are almost identical and also the

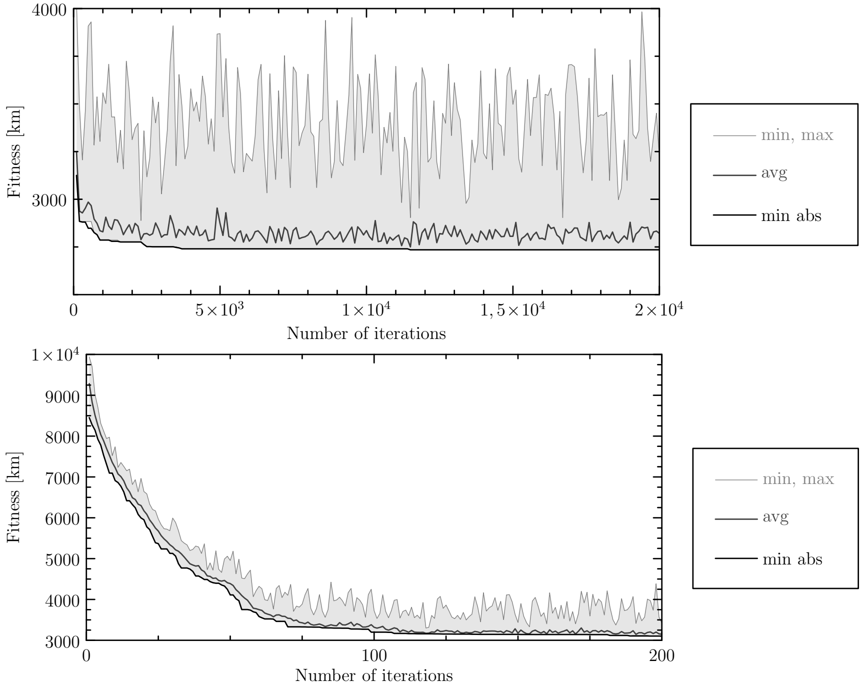


Fig. 6 *Algorithm progress – variant 2. For graph legend see Fig. 5*

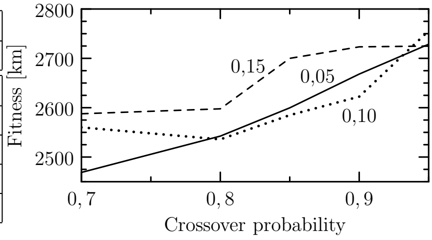
average fitness value has only small deviations. There is fast convergence in the beginning of the algorithm run (Fig. 6 bottom) and after five thousand steps there is almost no change in the found results (Fig. 6 top).

Variant 3 – alternating of both crossover methods without elitism

In the third variant the crossover operators from variant one and two are combined. The crossover method is chosen randomly in each step with a probability of 0.5. The minimum fitness function values are presented in Tab. IV.

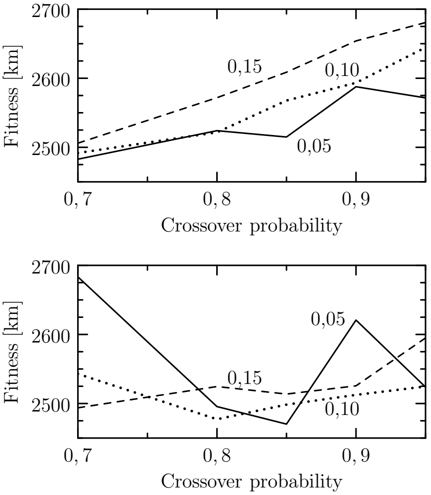
The results from the combination of both crossover types are even better than those from variant 2. The minimum total route length found of 2468.8 km is also shorter than the results obtained from the Clarke-Wright method (2492.1 km). The graph in Fig. 7 shows that, unlike the second variant, the decline in the fitness

Population size 20		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2468.8	2560.1	2587.7
	0.80	2542.4	2535.8	2597.7
	0.85	2599.7	2584.9	2700.0
	0.90	2668.1	2622.4	2723.3
	0.95	2728.6	2755.7	2724.9



Population size 30		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2482.8	2491.8	2506.0
	0.80	2524.1	2521.7	2571.8
	0.85	2514.9	2567.7	2608.7
	0.90	2587.7	2593.4	2654.0
	0.95	2571.6	2644.6	2680.6

Population size 60		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2683.3	2542.9	2493.8
	0.80	2495.6	2477.2	2524.6
	0.85	2470.2	2498.4	2513.7
	0.90	2620.8	2512.5	2525.6
	0.95	2523.8	2524.7	2594.9



Tab. IV *Minimum fitness function values for variant 3 – for legend description see Tab. II*

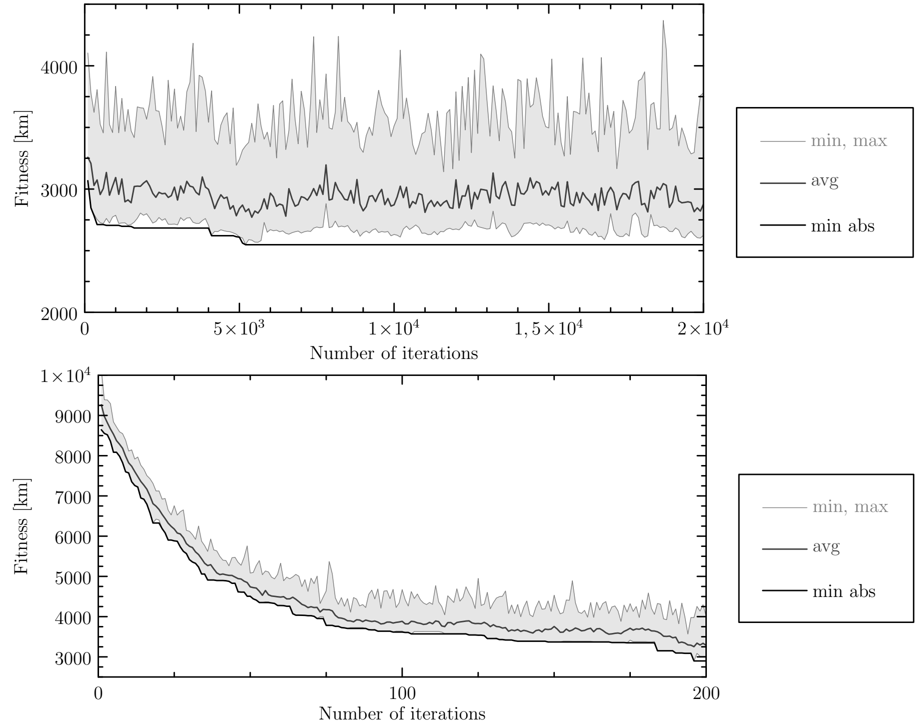


Fig. 7 *Algorithm progress – variant 3. For plot legend see Fig. 5*

function is slower at the beginning and also during the following progress the minimum and average values fluctuate more. A higher variability probably enabled better results to be found. The results do not show any significant dependence on the initial parameters as can be seen in Tab. IV.

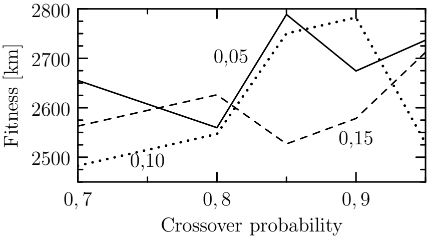
Variant 4 – random insert crossover with elitism

In variant 1 we came to the conclusion that not including elitism into the algorithm can often cause good solutions to be lost, and therefore lead to worse overall results. In this variant the crossover type from variant 1 is kept and in addition elitism is included.

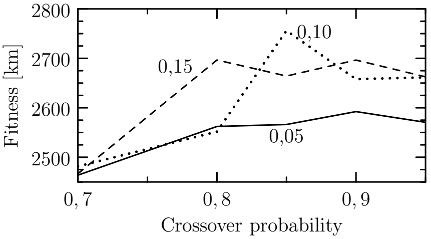
Even though the change in algorithm is relatively small, the final results changed rapidly and there is a considerable improvement in the found fitness value. The results are similar to those in variants 2 and 3, the best value found this time is 2463.7 km. The summary of the fitness values found is given in Tab. V.

With regard to the use of random crossover, the decline of the best fitness value is slower, as was expected, when compared against the previous variants and also the variability of the population is greater as can be observed in Fig. 8.

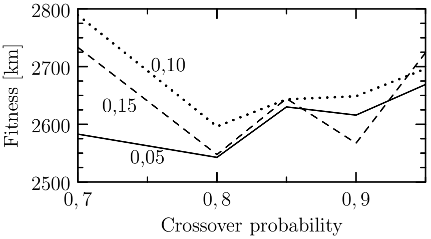
Population size 20		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2655.2	2482.7	2563.0
	0.80	2559.7	2546.9	2626.2
	0.85	2788.6	2750.2	2526.9
	0.90	2674.4	2782.5	2578.4
	0.95	2736.7	2528.4	2712.2



Population size 30		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2463.7	2481.5	2466.7
	0.80	2562.2	2551.7	2696.6
	0.85	2566.2	2756.0	2664.2
	0.90	2592.3	2658.0	2696.5
	0.95	2570.6	2661.5	2662.6



Population size 60		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2582.9	2789.3	2733.3
	0.80	2542.6	2596.2	2547.5
	0.85	2630.1	2643.3	2644.6
	0.90	2615.9	2648.6	2567.2
	0.95	2669.0	2696.3	2724.1



Tab. V *Minimum fitness function values for variant 4 – for legend description see Tab. II*

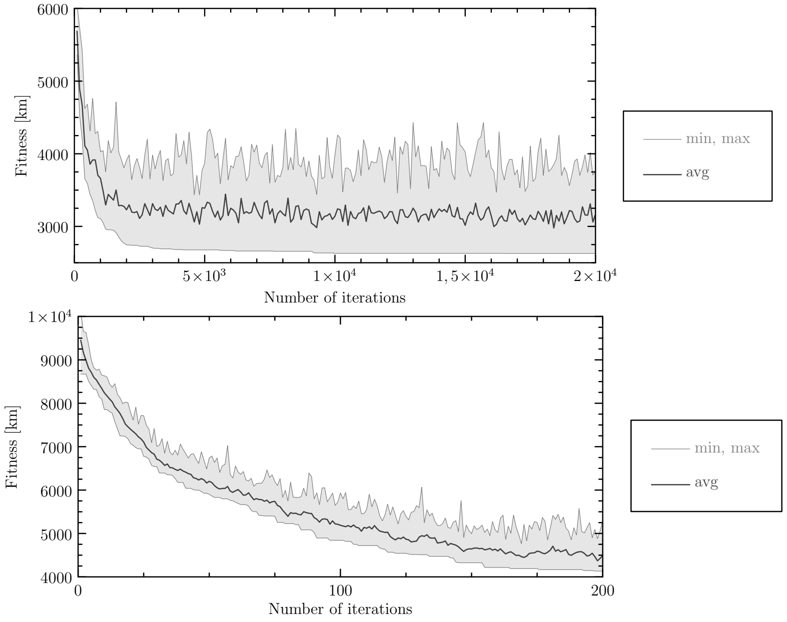


Fig. 8 *Algorithm progress – variant 4 – For graph key see Fig. 5. The graphs in Fig. 8, 9 and 10 do not display the absolute minimum of the fitness function which in the case of including elitism equals to the minimum of individual populations.*

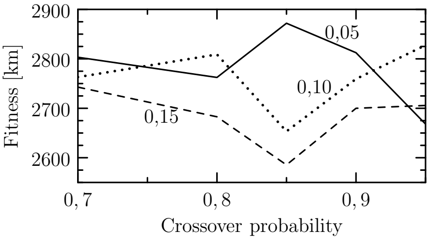
Variant 5 – nearest point crossover with elitism

Adding elitism into variant 2 has, unlike variant 4, only a small influence. It corresponds to the fact that the minimum of the population and absolute minimum in variant 2 without elitism were almost identical for most of the time and also the average fitness value of the population was only little higher than its minimum. After fast initial convergence there occur only minor changes during crossover which is to a certain extent deterministic. Therefore the best solution is transferred to the next generation with high probability even without elitism.

The minimum value found is 2468.4, as shown in Tab. VI together with the results of other combinations of probabilities.

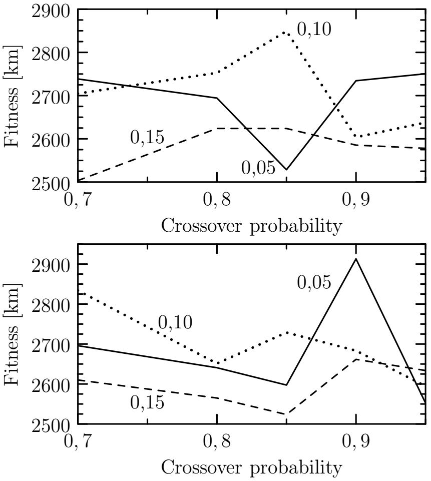
In Fig. 9 there can be seen a surprising decline in the minimum fitness function in the area around 10 000 iterations. It is probable that this is caused by the

Population size 20		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2803.2	2763.1	2742.8
	0.80	2762.4	2808.9	2682.8
	0.85	2872.0	2653.3	2585.5
	0.90	2812.4	2759.3	2700.0
	0.95	2668.1	2827.6	2705.7



Population size 30		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2738.5	2704.8	2503.4
	0.80	2694.3	2752.6	2624.0
	0.85	2528.5	2849.0	2623.9
	0.90	2734.4	2603.7	2585.4
	0.95	2750.5	2637.7	2578.0

Population size 60		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2695.9	2834.1	2610.0
	0.80	2640.7	2651.3	2564.9
	0.85	2597.3	2728.7	2523.6
	0.90	2913.1	2683.4	2661.6
	0.95	2552.4	2594.5	2633.7



Tab. VI *Minimum fitness function values for variant 5 – for legend description see Tab. II.*

influence of mutation, which ensured improvement after a high number of steps. This type of crossover makes little use of random processes and does not necessarily lead to the best results, which agrees with our assumptions.

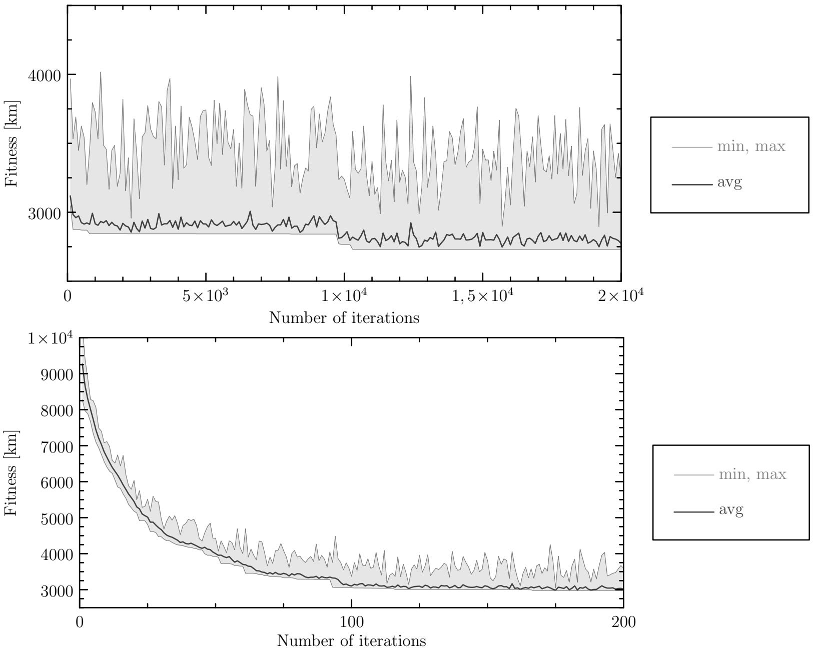


Fig. 9 *Algorithm progress – variant 5. For plot legend see Fig. 8.*

Variant 6 – alternating of both crossover methods with elitism

The last tested variant combines both types of crossover together with elitism. This enabled not only to find the lowest fitness value of all the tested variants, but also for all combinations of parameters the values were relatively low as can be seen in Tab. VII. The minimum fitness values vary from 2462.2 km to 2661.9 km. Moreover, there is a fast decline in the best fitness value at the beginning of the program run when the best value found does not change after 1000 steps (see graphs in Fig. 10).

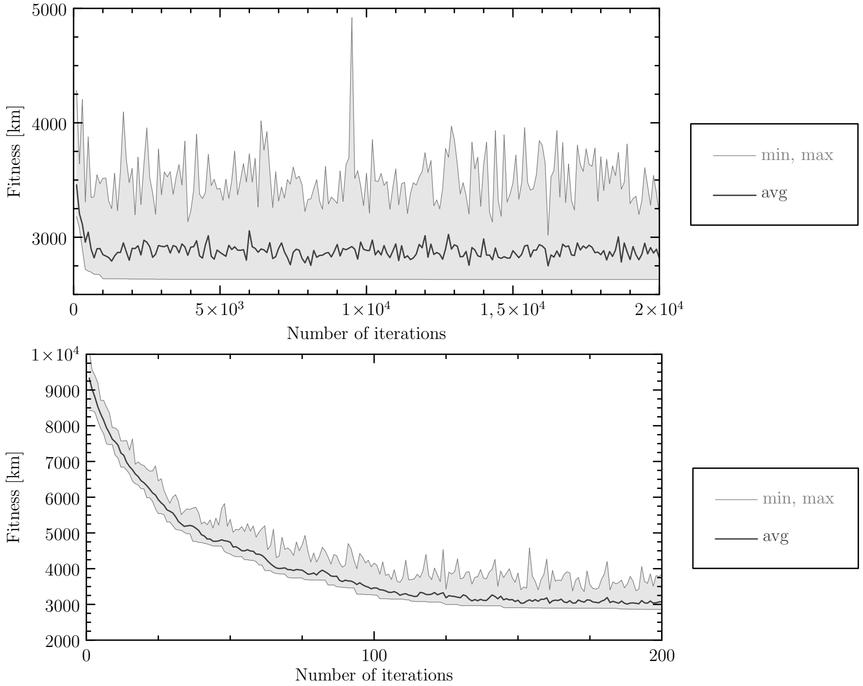
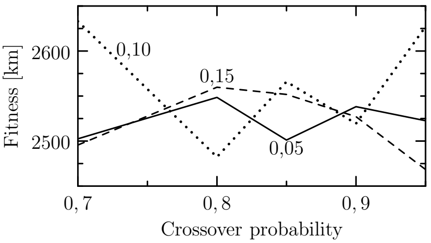
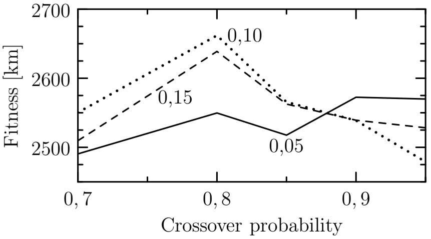


Fig. 10 *Algorithm progress – variant 6. For plot legend see Fig. 8*

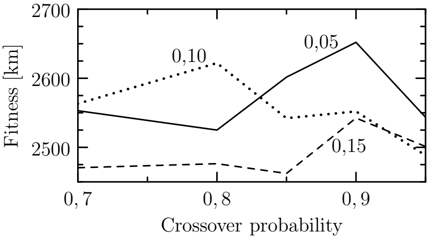
Population size 20		Mutation probability		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2502.5	2633.0	2495.6
	0.80	2548.5	2482.3	2559.8
	0.85	2501.1	2565.9	2551.8
	0.90	2538.2	2519.4	2527.1
	0.95	2522.6	2627.0	2468.4



Population size 30		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2490.5	2550.4	2509.6
	0.80	2549.7	2661.9	2638.8
	0.85	2517.6	2565.6	2562.6
	0.90	2572.5	2538.8	2539.2
	0.95	2569.9	2479.0	2528.3



Population size 60		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	2553.0	2563.2	2470.5
	0.80	2525.0	2622.0	2476.3
	0.85	2601.6	2542.2	2462.2
	0.90	2652.0	2552.0	2542.4
	0.95	2543.6	2488.4	2501.2



Tab. VII *Minimum fitness function values for variant 6 – for legend description see Tab. V*

All 6 variants results comparison

Comparison of the progress of all 6 variants is displayed in Fig. 11 and 12. The graphs depict the minimum fitness value found over a period of 20 000 iterations (Fig. 11), or, as relevant, for the first 200 steps (Fig. 12), equally as they were displayed for individual variants above.

The slowest decrease in the fitness value in the variant 1 can be observed in the figures and also the overall behaviour of this variant over time shows significantly higher values of the minimum fitness function than is the case of the other variants. The slower decrease can be seen also in variant 4 but after approximately two thousand iterations the values in variants 2 to 6 oscillate in a similar range and the final results do not vary much.

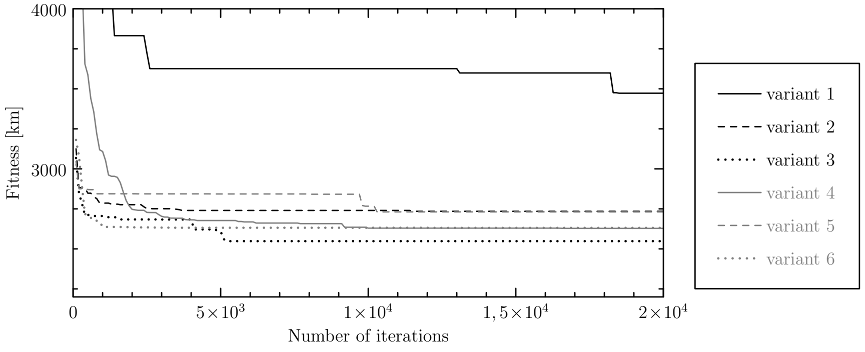


Fig. 11 *Comparison of the minimum values found of the overall algorithm progress for all 6 variants with initial setting of 30 individuals in the population, mutation probability 0.1, crossover probability 0.8 and 20 000 iterations.*

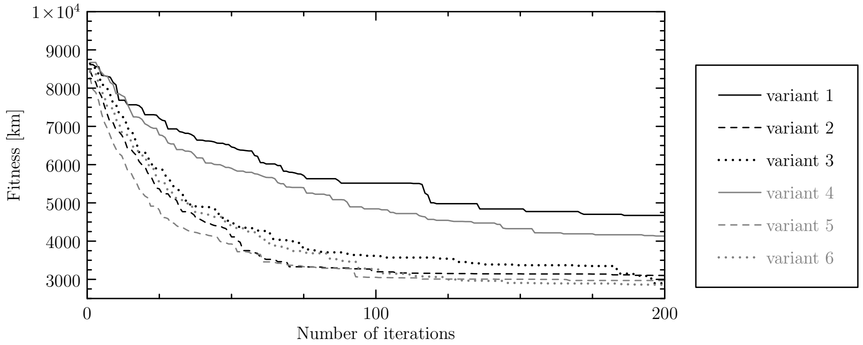


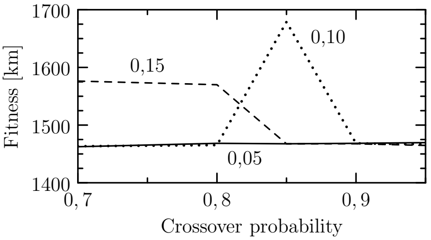
Fig. 12 *Comparison of the minimum values found for the first 200 steps of the algorithm for all 6 variants with the initial setting of 30 individuals in the population, mutation probability 0.1 and crossover probability 0.8.*

In variant 6 the results also had the smallest range of values. The variance of the results is important for general usage of the algorithm to ensure that the obtained result is not too far from the optimum. For this purpose it is convenient to use more than one combination of parameters.

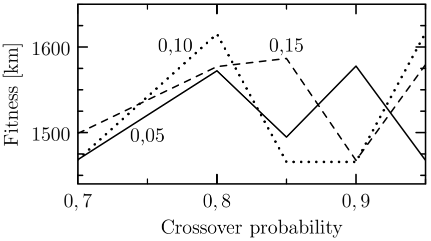
Depot Brno

On the basis of the behaviour of the individual variants described above, only the parameter setting of variant 6, i.e. that with the best results, was chosen for obtaining distribution routes from the Brno depot. This variant combines both types of crossover and includes elitism. The results are summarized in Tab. VIII, including their graphs. The minimum fitness function value found is 1462.5 km.

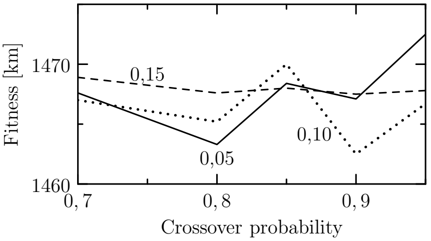
Population size 20		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	1462.5	1463.3	1576.3
	0.80	1468.4	1465.3	1570.0
	0.85	1467.6	1678.8	1467.5
	0.90	1468.4	1468.4	1467.6
	0.95	1469.6	1466.7	1464.8



Population size 30		**Mutation probability**		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	1468.4	1468.4	1499.0
	0.80	1572.2	1615.3	1577.1
	0.85	1494.7	1465.7	1586.6
	0.90	1577.7	1465.7	1467.5
	0.95	1467.5	1616.7	1579.4



Population size 60		Mutation probability		
		0.05	**0.10**	**0.15**
Crossover probability	**0.70**	1467.6	1467.0	1468.9
	0.80	1463.3	1465.2	1467.6
	0.85	1468.4	1470.0	1468.0
	0.90	1467.1	1462.5	1467.5
	0.95	1472.5	1466.7	1467.8



Tab. VIII *Minimum fitness function values for routes from depot Brno*

3.4 Discussion of genetic algorithm and Clarke-Wright algorithm results

If we compare the solutions obtained for both depots by the genetic algorithms against the Clarke-Wright method, we find that we managed to find routes shorter by 3.56%. The overview of the solutions found by both methods and percentage savings of GA compared to CW are presented in Tab. IX.

The best solution for Prague depot was obtained from variant 6 with a total length of the routes of 2462.2 km, which is 1.2% less than the solution outcome found using the Clarke-Wright method.

	depot Prague	**depot Brno**	**total**
Clarke-Wright [km]	2492.1	1577.60	4069.7
GA [km]	2462.2	1462.5	3924.7
savings of GA to CW	1.20%	7.30%	3.56%

Tab. IX *Comparison of the solutions found with CW and GA*

Apart from these results, demands for algorithm design and adjustments to particular VRP should also be compared. Clarke-Wright algotithm is heuristic algorith designed specificaly for VRP, only adjustment in substitution of limiting conditions (distance and number of garages were used instead of capacity constrains) had to be made. On the other hand genetic algorithm gives only general frame and all three operators that form the algorithm have to be designed for the purposes of the particular problem. Also number of parameters (size of population, mutation probability, crossover probability, elitism) has to be chosen when different initial settings of these parameters can give significantly different results. Although better solution was found using GA, it is not ensured that GA will give better results for different VRP problems and various settings of these parameters should always be tested.

With regard to computational time, both methods work in polynomial time. Clarke-Wright algorithm starts with the matrix of distances which is of size N^2 and these distances have to be sorted, for each distance the saving $i - j$ is calculated and then each saving is tested for combining i and j in one route, where sorting process is most time demanding part of the algoritm. In GA computational time is given purely by ending condition, in our case by number of iterations. Although

20 000 was used in our testing, from the results can be observed that best values were found much sooner and number of iterations can be reduced considerably.

4. Conclusion

In this paper we designed and analyzed a genetic algorithm used for VRP. In order to compare the behaviour and results of individual methods we used an example of optimising a spare parts supply chain, consisting of two depots. The Clarke-Wright method, which is the best known heuristic method for VRP, was used to propose distribution routes and for the same task a genetic algorithm was proposed and implemented.

VRP features certain specifics, so it is not possible to use the standard genetic operators commonly used to solve other mathematic problems. The algorithm was proposed with several variants of operators and tested with various combinations of initial parameter settings.

When the simplest operators were used the results were not satisfactory, but after including further changes we managed to create a genetic algorithm able to find the solution comparable to or better than the Clarke-Wright method. Based on the GA testing, the recommended values of initial parameters were found. However, the sensitivity of the best working variant to these parameters is not high and so relatively good results may be achieved with any initial setting. The initial parameter setting generally represents a problematic issue in the metaheuristic methods and so it is not quite clear how good or bad the results we get are. Moreover, thanks to randomness we can get different results with the same initial settings.

Acknowledgement

This paper was supported by research program No. MSM 68407700/43 “Development of Methods for Optimization in Transport System Operations”.

References

[1] Desrosier J., Dumas Y., Solomon M. M., Soumis F.: Time constraint routing and scheduling, in Handbooks in Operations Research and Management Science, Vol. 8, Network Routing, edited by M. O. Ball, T/L Magnanti, C. L. Monma, G. L. Nemhauser (eds.). Elsevier Science Publishers: Amsterdam, pp. 35–139, 1995.

[2] Psaraftis H. N.: Dynamic vehicle routing problems, in Vehicle Routing: Methods and Studies 16, pp: 223–248, 1988.

[3] Braysy O., Gendreau M.: Vehicle routing problem with time windows, Part 1: Route construction and local search algorithms, in SINTEF Applied Mathematics Report, Department of Optimization, Norway, 2001.

[4] Toth P., Vigo D.: The Vehicle Routing Problem, Philadelphia, Siam, 2001.

[5] Ropke S., Cordeau J.-F., Iori M., Vigo D.: Branch-and-Cut-and-Price for the Capacitated Vehicle Routing Problem with Two-Dimensional Loading Constraints, in Proceedings of ROUTE, Jekyll Island, 2007.

[6] Clarke G., Wright J. W.: Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. Operations Research, vol. 12, pp. 568–581, 1964.

[7] Glover F.: Heuristics for Integer programming Using Surrogate Constraints. in Decision Sciences 8, pp. 156–166, 1977.

[8] Reimann M., Doerner K., Hartl R. F. Analyzing a Unified Ant System for the VRP and Some of Its Variants, in EvoWorkshops 2003, pp. 300–310, 2003.

[9] Czarnas P., Czech Z. J., Gocyla P.: Parallel Simulated Annealing for Bicriterion Optimization Problems, in PPAM 2003, pp. 233–240, 2004.

[10] Jaszkiewicz A., Kominek P.: Genetic Local Search with Distance Preserving Recombination Operator for a Vehicle Routing Problem, in European Journal of Operational Research, vol. 151, pp. 352–364, Elsevier, 2003.

[11] Kelly J., Xu J. P.: A Network Flow-Based Tabu Search Heuristic for the Vehicle Routing Problem, in Transportation Science, vol. 30, pp. 379–393. 1996.

[12] Pereira F. B.: GVR: a New Genetic Representation for the Vehicle Routing Problem [online]. Available: http://fmachado.dei.uc.pt/wp-content/papercite-data/pdf/ptmc02a.pdf

[13] Beasley D., Bull D. R., Martin R. R.: An overview of genetic algorithms: Part 1, fundamentals. University Computing, vol. 15-2, pp. 58-69, 1993.

[14] Rybickova A.: Analyza a optimalizace zasobovani autorizovanych servisu vozidel Peugeot a Citroen. Bakalarska prace, CVUT v Praze Fakulta dopravni, 2010.

[15] Rybickova A.: Praha: Vyuziti genetickych algoritmu vulohach diskretni optimalizace. Diplomova prace, CVUT v Praze Fakulta dopravni, 2012.