Analysis of distance between vehicle routing problem solutions generated by memetic algorithms

Marek Kubiak[1]

[1] Poznan University of Technology, Institute of Computing Science, Poznań, Poland,
e-mail: Marek.Kubiak@cs.put.poznan.pl

Abstract. This paper focuses on analysis of distance between solutions of the capacitated vehicle routing problem generated by memetic algorihms with different crossover operators. The goal of the analysis is to see what are relative positions of such solutions in search spaces and if different algorithms explore different parts of these spaces. In the described memetic algorithms five different crossover operators are used. The conducted computational experiment shows that solutions generated by the algorithms have very similar, very good quality. From the distance analysis it appears that there are two types of instances of the considered problem: 10 instances have very similar solutions, concentrated in small regions of spaces ('big valleys'), while other 12 have good solutions which reside in different parts of search spaces, implying wide and flat regions of good solutions.

1 Introduction

When an evolutionary algorithm is to be used in combinatorial optimisation there is usually a design issue to be faced: the construction of crossover operators for the considered problem. Very often, the well-known, simple operators constructed for binary or continuous optimisation (such as single- or multiple-point crossover, uniform crossover, etc.) are not useful; they yield poor results because they do not consider specific properties of solutions of combinatorial problems. It may be seen, for example, in the case of the common test-bed for metaheuristics, the travelling salesperson problem (TSP), where one of the most efficient operators, edge-assembly crossover (EAX, see [19]) was designed especially for this task after many years of research.

One of possible approaches to this issue of operator design is to rely mainly on intuition and experience of a designer. Another one is to base it also on properties of the solution space to be searched. The method of systematic construction of recombination operators based on a property of search spaces called fitness-distance correlation ([5], [6], [9]) is of the second type. In case the fitness-distance correlation indicators are found, and the so-called 'big-valley' ([1], [7], [9], [11]) structure exists, the method leads to the construction of distance preserving crossover operators (DPX), described e.g. by Merz in [11], [13]. This work indicated that such operators had been extremely efficient when used in memetic algorithms (MAs) solving many classical

[1] The work of the author was supported by State Committee for Scientific Research (KBN) through research grant no. 3 T11F 004 26.

combinatorial problems: TSP, quadratic assignment (QAP), graph bi-partitioning (GBP), binary quadratic programming (BQP).

Recently, the author performed the fitness distance analysis and systematic construction of recombination operators for the capacitated vehicle routing problem (CVRP, [9], [10]) and proposed 4 operators for the problem: 3 crossovers and 1 mutation. All these operators were based on notions of distance between solutions and preserved related distances completely or to very high extent.

The main purpose of this text is to analyse the distance and relative position of good solutions generated by memetic algorithms which differ only in crossover operators (those systematically constructed and also those taken from literature). The relevant research questions are: are these solutions concentrated in some area of the search space or rather spread all over it? Are solutions of different algorithms (crossover operators) separated in the search space or mixed? What is the relationship between the quality of such solutions and their relative position in the solution space? Answers to these questions might provide some insight into the nature and existence of the phenomenon of 'big valley' in the CVRP.

The author has not yet seen such analysis in literature: focusing on distance between solutions, rather than on quality (fitness). Here, the distance is understood as a measure of difference between properties of solutions and these properties have no direct connection with quality. Indeed, the quality of solutions does not even have to be known to examine their distance.

2 The capacitated vehicle routing problem (CVRP)

The goal of this problem is to plan delivery of goods from a transportation company's depot (headquarters) to geographically distributed customers. This plan has to take into consideration all the customers and their specific demands for the goods. The company owns a set of vehicles and dispatches them to customers on routes defined by the plan. All vehicles are identical and must not be loaded for delivery with more goods than the specified maximum capacity. During deliveries a vehicle arrives at locations of subsequent customers (as specified in its route plan), unloads the demanded amount of goods and continues its itinerary. Finally, it returns to the depot. The cost of a solution (a delivery plan, which is a set of disjoint routes) is defined as the sum of lengths of all routes; it should be minimized.

The reader interested in more detail on the CVRP is referred e.g. to [16]. This work is worth reading and it also contains many useful references to other publications.

The instances of the CVRP considered here are the well-known 22 ones proposed by Christofides (7 data files with prefix 'c' in names), Taillard (13 files, prefix 'tai') and Fisher (2 files, prefix 'f') (the same instances were used in [9], [10]; they are described in [16]). All these examples of the CVRP were attempted to be solved by many researchers and by means of diverse algorithms. The best-known solutions were generated by a tabu search algorithm of Rochat and Taillard ([16]), which also contained additional diversification and intensification mechanisms. That paper is the source of quality of the best-known solutions for this work.

Examples of solutions for one instance of the CVRP (c50, with 50 customers) are shown in Figure 1. Each of the solutions consists of a depot (the centrally located circle) and customers (the other circles). All routes (lines connecting circles) start at the depot, pass through at least one customer, and return to the headquarters (lines to and from the depot are half-cut in order not to obscure the images).

3 A note on the fitness-distance analysis of the CVRP

Fitness distance analysis (FDA) of the CVRP conducted in the past (see [9] and [10]) required that measures of distance between solutions of the problem be defined. The author defined 3 such measures (normalized distance metrics):

- d_c perceives solutions as clusters (groups) of vertices (customers) which are put in the same route; the value of distance reflects the largest difference between clusters;
- d_{pn} also considers solutions as clusters only (like d_c), but the value of distance indicates the average difference between clusters, instead of extremes;
- d_e takes into account the edges of solutions; the value of distance reflects the number of different edges in them (similarily to the measure used in [1] to compare solutions of the TSP).

The results of the FDA showed that the strongest correlation between quality and distance was obtained when d_e was the distance metric used, d_{pn} being slightly worse and d_c the worst. Although values of the linear determination coefficient (R^2) indicated weak correlation, they were high enough to become significant with respect to the classification introduced by Jones and Forrest in [7] (correlation coefficient r>0,15), who described problems with such values of correlation as easy for optimisation.

These very distances, d_e and d_{pn}, provided the author with the basis for construction of the distance preserving crossover operators (see [9]): CPX (preserving d_{pn}), CEPX (preserving d_e) and CECPX (preserving both d_{pn} and d_e). These operators, some with amendments, are also used in this work. The mentioned distance metrics are applied here to analysis of distance between solutions of memetic algorithms.

4 Crossover operators for memetic algorithms

The core of the MAs used in this work, and the only difference between them, are crossover operators.

4.1 The operators taken from literature

The author implemented two such operators, route-based crossover (RBX) and split crossover (SPX), taken from descriptions of genetic algorithms for vehicle routing problems found in literature.

The route-based crossover ([14], [8]) creates an offspring by copying a set of routes from one parent and completing it with routes from another one. The number of routes taken from the first parent is random and the copied ones are selected with uniform probability. Then, all routes from the second parent are considered for copying into the offspring and subsequent customers of each route are put in the offspring only if they are not assigned to any route yet.

The author's implementation of RBX differs slightly from the original: the number of routes copied from the first parent is not constant (one) but random, and the complicated repair procedure applied to each offspring in [14] is not used here, since it does not ensure the creation of a feasible solution.

An exemplary offspring of RBX and its parents are shown in Figure 1. The routes inherited from the first parent (parent$_1$) are emphasized in grey.

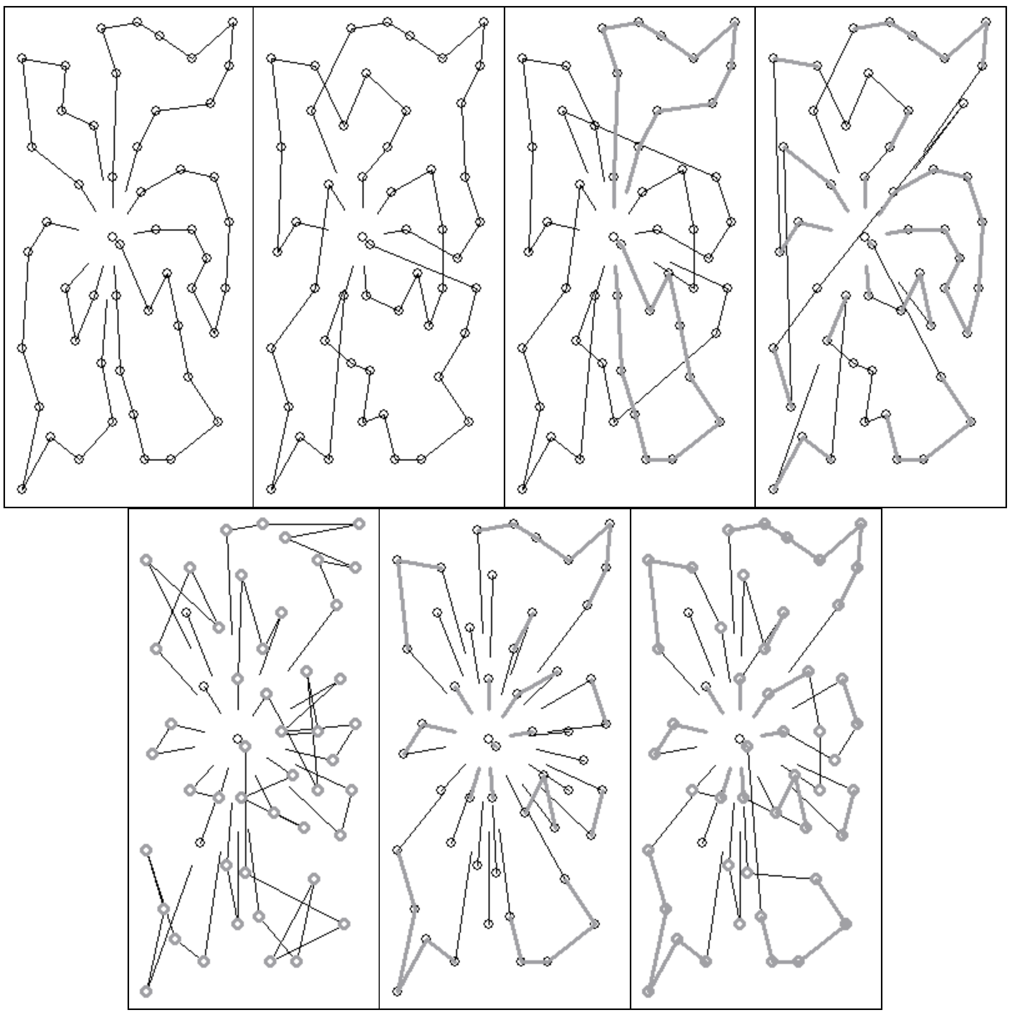

Figure 1. An example of parents (local optima) and offsprings for instance c50.
Top, left to right: parent$_1$, parent$_2$, offsprings of RBX and SPX.
Bottom, left to right: offsprings of CPX2, CEPX, CECPX2.

Prins's split crossover, SPX ([15]), is based on a completely different idea, that is, of perceiving a CVRP solution as a sequence of customers, just as in the case of the TSP; the division into routes is ignored. Thus, the encoding of solutions in the Prins's genetic algorithm is based on sequences. A deterministic decoding mechanism is used to obtain a valid CVRP solution from a sequence of customers. This decoding mechanism has a very strong property: it always provides the minimum-cost set of routes for the given sequence. Therefore, the CVRP problem is reduced to the one of finding an optimal sequence (permutation) of customers. Consequently, the SPX is simply the order crossover operator (OX) usually applied to permutations. The fitness of an offspring is obtained through the decoding procedure.

One such decoded offspring of SPX is shown in Figure 1. Due to the mentioned encoding change it may contain extraordinarily long edges, which are seen in the picture. The emphasized routes and edges are inherited from parent$_1$.

4.2 The systematically constructed operators

The first operator designed by the author ([9]) was called the clusters-preserving crossover (CPX), because it preserved the distance metric comparing clusters of customers in solutions, d_{pn}. Its current version (CPX2) is also based on this distance: it recognizes clusters of customers which are put in one route in both parents (the common clusters) and creates routes in an offspring by a random choice of edges in these clusters. Eventually, the offspring preserves all clusters common to both parents. The customers outside those clusters are assigned to one-customer routes.

The CPX2 offspring shown in Figure 1 has the customers in common clusters emphasized. Unfortunately, the distinction between clusters would require colour images to be used, but one may note that one cluster means one route, so different clusters may be recognized.

The second operator, called the common-edges-preserving crossover (CEPX) is based on d_e, the distance in terms of edges. It computes the set of edges common to both parents, which sometimes form long paths, and creates offspring's routes using only these edges to connect customers; some additional depot-customer edges may be added during this process to start and finish routes. Finally, the customers which are not the end of any common edge are put in one-customer routes.

An example of a CEPX offspring is included in Figure 1; edges common to parents are emphasized.

The last crossover operator described in [9] was called the clusters-and-common-edges-preserving crossover (CECPX), since it preserved both d_{pn} and d_e. Its current version (CECPX2) also preserves these metrics, and is in a way an amalgamation of CPX2 and CEPX. First, it finds in parents the sets of common clusters and common edges. Then, it creates each route in an offspring by using common edges to connect customers within one cluster; if there is no common edge to connect vertices in this cluster then a randomly chosen edge is added in order not to divide the cluster. Finally, customers outside common clusters and edges are put in one-customer routes.

One such offspring is presented in Figure 1; common clusters and edges are drawn in grey.

The only difference in pairs CPX2-CPX and CECPX2-CECPX is in the way common clusters are computed: the previous versions did not ensure inclusion of all common clusters in offsprings, while the current ones make sure no such clusters are omitted.

5 Description of the experiment with memetic algorithms

In order to compare the quality and distance of solutions generated by different memetic algorithms a large computational experiment was conducted. Five MAs were run, each one with only one crossover operator (CPX2, CEPX, CECPX2, RBX, SPX; these are also short names of the described algorithms). All algorithms were run 36 times for each instance of the CVRP. Each run was independent from others and employed the same settings:

- the steady-state type algorithm;

- initial population: one Clarke and Wright solution ([2]), several solutions generated by the Gillet and Miller heuristic ([4]), random solutions; full greedy local search on all initial solutions before a process of evolution is run;
- one crossover and one mutation attempt (CPM, clusters-preserving mutation, see [9]) in one generation;
- full greedy local search after each successful crossover and mutation (i.e. a feasible offspring);
- local search based on aggregated neighbourhoods of 3 operators: merge of any 2 routes, exchange of any 2 edges, exchange of any 2 customers;
- an offspring is accepted to the population only if it is better than the worst solution currently in the population and has different value of quality than all others in the population;
- population size: 30;
- uniform selection;
- stop after 120 generations without a change in the whole population.

The steady-state memetic algorithms with full local search introduces very strong selection pressure during artificial evolution, therefore the selection procedure used equal probabilities for all members of each population in order not to make the pressure stronger. The goal of the stopping criterion was to force all algorithms to completely converge and the process of evolution to fade-out. Thus, the times of computation for each algorithm and run were different, with the slowest (CPX2) being several times slower than the fastest (RBX).

These settings for the memetic algorithms were not determined in any special manner. On the contrary, the author set them only to some values he deemed sensible, because it is not the author's intention to perform intensive tuning of parameters of algorithms; he thinks such tuning is a way of transferring the effort of design from an algorithm to values of such parameters and should be avoided whenever possible.

All the memetic algorithms were run on 12 identical PCs, each with Intel Pentium 4 3.2 GHz processor, 1GB RAM, running Windows XP. Total time of computation for this experiment exceeded 82 days.

6 Analysis of results

6.1 Quality of solutions

Table 1 shows basic statistics on the quality of solutions: the average quality above the best-known solution and the quality of the best solution found in all 36 runs of each algorithm. It may be noted that differences between algorithms are very low: maximum difference between averages is 1,13%, while on average it is only 0,2%. Nevertheless, some of these differences are significant, because the deviations of quality (not shown) are extremely small.

Table 2 presents aggregated results of comparison of these averages for each instance by means of the U test (the U statistic is normally distributed under the null hypothesis). Each entry in this table shows how many times (i.e. for how many instances) the algorithm in a row is better than the one in a column. The two-sided version of the test was used with the level of significance set to 0.05 (5%). The last column in this table (the difference between sum of the rows and the sum of the columns for an algorithm) presents the overall quality indicator for each algorithm (the higher, the better, because it means an algorithm won more direct comparisons with other ones). Consequently, the best algorithm with respect to the quality of solutions is CEPX; SPX, CECPX2, RBX and CPX2 follow in the ranking (in that order).

Table 1. Basic statistics on the quality of solutions of each memetic algorithm and for each instance. The smallest average values for each instance are emphasized in bold.

Instance	Best-known	CPX2		CEPX		CECPX2		RBX		SPX	
		Avg. [%]	Best [%]	Avg. [%]	Best [%]	Avg. [%]	Best [%]	Avg. [%]	Best [%]	Avg. [%]	Best [%]
c100	826.14	0.40	0.15	0.36	0.15	0.36	0.00	0.33	0.00	**0.31**	0.15
c100b	819.56	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
c120	1042.11	0.12	0.00	**0.06**	0.00	0.07	0.00	0.07	0.00	**0.06**	0.00
c150	1028.42	0.89	0.12	**0.54**	0.20	0.66	0.26	0.95	0.00	0.83	0.17
c199	1291.45	1.62	0.68	**0.99**	0.26	1.28	0.19	1.50	0.73	1.66	0.49
c50	524.61	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
c75	835.26	0.26	0.00	0.28	0.00	**0.22**	0.00	0.39	0.00	0.23	0.00
f134	11629.60	0.03	0.00	0.02	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
f71	241.97	**0.00**	0.00	0.01	0.00	**0.00**	0.00	0.06	0.00	**0.00**	0.00
tai100a	2041.34	1.37	0.32	**1.29**	0.35	1.46	0.32	1.57	1.48	1.40	0.32
tai100b	1940.61	0.04	0.00	0.02	0.01	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
tai100c	1406.20	0.45	0.00	**0.25**	0.00	0.45	0.00	0.64	0.24	0.53	0.02
tai100d	1581.25	1.02	0.72	1.02	0.99	0.98	0.00	0.99	0.31	**0.92**	0.00
tai150a	3055.23	0.15	0.00	**0.09**	0.00	**0.09**	0.00	0.16	0.00	**0.09**	0.00
tai150b	2727.77	0.22	0.00	**0.16**	0.00	0.30	0.00	0.41	0.00	**0.16**	0.00
tai150c	2341.84	1.24	0.89	**0.99**	0.72	1.09	0.85	2.12	0.91	1.58	0.84
tai150d	2645.39	0.89	0.37	**0.80**	0.03	0.86	0.52	0.85	0.22	0.82	0.05
tai385	24431.44	1.07	0.67	0.79	0.49	0.84	0.49	0.88	0.49	**0.74**	0.40
tai75a	1618.36	0.12	0.00	**0.01**	0.00	0.02	0.00	0.03	0.00	0.04	0.00
tai75b	1344.64	0.03	0.00	0.02	0.00	0.03	0.00	**0.01**	0.00	**0.01**	0.00
tai75c	1291.01	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
tai75d	1365.42	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
Avg.[%]	-	0.45	0.18	**0.35**	0.14	0.40	0.12	0.50	0.20	0.43	0.11

Despite the results of the statistical tests the author deems the differences negligible from the point of view of the overall quality of the obtained solutions: they are all of very good quality (near to the best-known ones). Therefore, it is worth checking if these solutions are also near each other in the sense of distances d_{pn} and d_e, or perhaps the algorithms converge to different regions of the search space.

Table 2. Aggregated results of the U test for significance of difference in averages. One entry indicates how many times (for how many instances) the null hypothesis was rejected in favour of the alternative one, i.e. how many times the algorithm in a row was better than the one in a column

Alg_{row} better than $\text{Alg}_{\text{column}}$	CPX2	CEPX	CECPX2	RBX	SPX	Sum (row)	Sum (row) – Sum (column)
CPX2	0	0	0	4	1	5	-30
CEPX	10	0	4	6	4	24	16
CECPX2	8	2	0	6	3	19	11
RBX	7	2	1	0	0	10	-11
SPX	10	4	3	5	0	22	14
Sum (column)	35	8	8	21	8		

Table 3. Information about new best solutions found.

Instance	Best-known	New best	Algorithms
tai100b	1940.61	1940.379	CEPX
tai150b	2727.77	2727.669	SPX, RBX
tai75b	1344.64	1344.619	CPX2, CEPX, CECPX2, SPX, RBX

As a side effect of the conducted computational experiment some new best solutions were found for 3 instances of the CVRP. Basic information on these solutions and the algorithms which generated them is shown in Table 3[2]. It may be seen that the differences between new and old best solutions are extremely small.

6.2 Distance between solutions

All the solutions generated by the memetic algorithms were also examined from the point of view of the mentioned distances; Table 4 provides statistics on d_e and Table 5 on d_{pn}.

Distance in terms of edges: d_e. Table 4 contains 11 columns. Column 2 (All edg.) presents the total number of edges in a good solution of each instance. Columns 3 and 4 are cited after [10] and show determination coefficients (r^2) between fitness and distance, and average distance (Avg. d_e) between solutions in large sets of random local optima (RLO) generated for fitness-distance analysis. Column 5, computed from values in column 4, shows the average percentage of different edges in RLO. Column 6 contains numbers of different solutions generated in all 180 independent runs of all 5 MAs. Next in this table are average (Avg. d_e, column 7) and maximum (Max. d_e, column 8) distances in pairs of solutions generated by MAs, and twice the height of

[2] The contents of the new best solutions found in this experiment are available at: http://www.cs.put.poznan.pl/mkubiak/research/cvrp

clustering trees generated for all sets of solutions (column 9, the meaning of these trees is explained later in the text). Finally, column 10 (Avg. diff. edg.) presents the average number of different edges in two solutions of MAs, while in column 11 these numbers are translated into percentages of all edges (as calculated in column 2).

Table 4. Comparison of solutions generated by memetic algorithms with distance d_e.

Instance	All edg.	r^2 ([10])	Avg. d_e ([10])	Avg. diff. edg. [% All edg.]	Gen. diff. sol.	Avg. d_e	Max. d_e	2*tree-height	Avg. diff. edg.	Avg. diff. edg. [% All edg.]
1	2	3	4	5	6	7	8	9	10	11
c100	108	0.197	0.673	50.7%	65	0.317	0.650	0.561	20	18.8%
c100b	110	0.347	0.540	37.0%	6	**0.035**	0.120	0.087	**2**	**1.8%**
c120	127	0.020	0.714	55.5%	40	**0.182**	0.432	0.317	**13**	**10.0%**
c150	162	0.296	0.679	51.4%	169	0.423	0.627	0.513	43	26.8%
c199	216	0.237	0.701	54.0%	180	0.543	0.679	0.604	80	37.3%
c50	55	0.167	0.605	43.4%	1	**0.000**	0.000	0.000	**0**	**0.0%**
c75	85	0.117	0.688	52.4%	49	0.473	0.750	0.620	26	31.0%
f134	141	0.160	0.633	46.3%	62	**0.094**	0.321	0.225	**7**	**4.9%**
f71	75	0.237	0.477	31.3%	5	**0.043**	0.370	0.221	**2**	**2.2%**
tai100a	112	0.083	0.595	42.3%	61	0.258	0.679	0.520	17	14.8%
tai100b	111	0.238	0.610	43.9%	29	**0.106**	0.577	0.476	**6**	**5.6%**
tai100c	111	0.246	0.625	45.5%	17	0.217	0.604	0.566	14	12.2%
tai100d	111	0.178	0.609	43.8%	39	0.259	0.580	0.500	17	14.9%
tai150a	165	0.009	0.627	45.7%	125	0.208	0.546	0.401	19	11.6%
tai150b	164	0.072	0.663	49.6%	126	0.327	0.645	0.539	32	19.5%
tai150c	165	0.190	0.629	45.9%	178	0.449	0.636	0.511	48	28.9%
tai150d	164	0.042	0.636	46.6%	140	0.291	0.604	0.479	28	17.0%
tai385	433	0.122	0.745	59.4%	180	0.439	0.606	0.524	122	28.1%
tai75a	85	0.165	0.621	45.0%	15	**0.112**	0.547	0.510	**5**	**5.9%**
tai75b	85	0.025	0.617	44.6%	17	0.182	0.534	0.422	9	10.0%
tai75c	84	0.187	0.581	40.9%	2	**0.004**	0.069	0.069	**0**	**0.2%**
tai75d	84	0.244	0.519	35.0%	6	**0.045**	0.320	0.300	**2**	**2.3%**
Avg.	-	0.163	0.627	45.9%	68.7	0.228	0.495	0.408	-	13.8%

First important observation is that, on average, MAs generate only 68,7 different solutions of the CVRP for the total of 180 possible. This number might be the first indicator of the fact that solutions of these algorithms are similar to each other. Of course, there are differences between instances: some have more than this average (like c150, c199, tai150a-d, tai385), whereas others have less (the smaller instances). Moreover, the number of different solutions is not enough to prove that solutions are clustered in the search space or spread all over it.

Further evidence for this conclusion is found while comparing distance d_e between RLO and between solutions of MAs. The values in columns 4 and 5 are provided here in order to facilitate comparison between these sets of solutions. Indeed, the comparison of columns 4 and 5 with corresponding columns 7 (or 8) and 11 indicates that solutions of MAs are always significantly more concentrated in the search space than random local optima; the difference between averages amounts to $\Delta d_e \approx 0{,}4$ and Δ(% different edges)≈32%. Also in this case there are differences between instances. For some of them (indicated in bold in column 7) the gain in the average distance is very high (either the gain is >0,5 or the value in column 7 is <0,05 itself). For the others the gain is lower and the maximum distance between solutions of MAs has very similar value to Avg d_e in the sets of local optima (column 4), so the size of the search space which contains solutions of MAs is still rather large. This division of the set of instances implies a cut-off value of 10% for the percentage of different edges (column 11): instances with this value lower than 10% have MAs solutions highly clustered, while the others still have significant differences between solutions perceived from the perspective of edges (tai75b is a borderline case). An example of the first type might be instance c120: there are, on average, only 13 different edges between solutions of MAs.

To summarize, it might be said that for 9 or 10 instances the solutions of MAs are highly clustered (from the point of view of d_e), which indicates 'narrow valleys' to be searched by optimisation algorithms based on preservation of edges (easy optimisation). In other cases (12 or 13 instances) this clustering of solutions is weaker, indicating 'wider valleys' and relatively harder optimisation. The hardest instances seem to be: c199, c75, tai150c, tai385 and c150 which have on average more than 20% of different edges in solutions of memetic algorithms.

Distance in terms of pairs of nodes: d_{pn}. Table 5 contains statistics similar to those in Table 4, but from the point of view of d_{pn}. It does not contain, however, the values of distance computed as the numbers of nodes differently assigned to clusters. The author attempted to compute it, but failed to find an appropriate formula. Therefore, only the statistics on the distance d_{pn} itself are reported.

Comparison of columns 3 and 4 implies conclusions similar to those formulated in case of d_e: for 10 instances solutions of MAs are significantly closer to each other (on average) than random local optima (the gain in d_{pn} is >0.5 or average d_{pn} itself is <0,05). Those instances are indicated in bold in column 4 (9 of them are also indicated in Table 4). In case of the other instances the gain in average d_{pn} is lower, which implies 'wider valleys' to be searched than in case of the 10 instances mentioned above (which should be easy in optimisation by cluster-based algorithms). Additionally, 4 instances should be extremely easy from this point of view (c100b, c50, f71, tai75c), due to very low average d_{pn} between solutions of MAs (<0,01).

UPGMA clustering trees of solutions. A clustering tree generated by the UPGMA method (see [17]) is a way of visualizing relationships between objects which are described by a distance matrix. Four such trees generated from matrices of distance between solutions of MAs are shown in Figure 2 and Figure 3.

In Figure 2 the trees are rendered horizontally, with root nodes on the left and leaf nodes on the right. Each leaf represents one CVRP solution generated by a certain algorithm; a label on the right describes the quality of a solution and the algorithm which produced it. Two nodes of each tree (not only leaves) are linked by a parabola on a certain distance level, as indicated on a ruler under each tree. This level of a link corresponds to the half of the average distance between linked elements (i.e. distance between solutions or clusters of solutions). If certain nodes are linked by vertical lines it means the distance between them is equal to zero. The height of a tree (the level of its root node) is approximately the half of the maximum distance between solutions;

it is usually slightly lower than this maximum (compare columns 8, 9 in Table 4 and 5, 6 in Table 5) due to the operation of averaging of distance performed while creating clusters of solutions. Additionally, all leaves in each tree are roughly sorted in ascending order of the objective function, from top to bottom (only if the structure of a tree allows such ordering to be made).

Table 5. Comparison of solutions generated by memetic algorithms with distance d_{pn}.

Instance	r^2 ([10])	Avg. d_{pn} ([10])	Avg. d_{pn}	Max. d_{pn}	2*tree-height
1	2	3	4	5	6
c100	0.126	0.731	0.416	0.774	0.703
c100b	0.468	0.589	**0.000**	0.000	0.000
c120	0.152	0.746	**0.104**	0.264	0.200
c150	0.232	0.759	0.486	0.740	0.634
c199	0.242	0.791	0.660	0.814	0.695
c50	0.173	0.659	**0.000**	0.000	0.000
c75	0.113	0.752	0.527	0.827	0.746
f134	0.045	0.709	**0.018**	0.251	0.204
f71	0.378	0.338	**0.003**	0.119	0.119
tai100a	0.113	0.690	0.267	0.800	0.653
tai100b	0.233	0.698	**0.080**	0.619	0.544
tai100c	0.380	0.707	0.296	0.678	0.594
tai100d	0.272	0.676	0.299	0.678	0.574
tai150a	0.000	0.731	**0.195**	0.654	0.504
tai150b	0.223	0.731	0.379	0.690	0.620
tai150c	0.217	0.736	0.578	0.767	0.674
tai150d	0.057	0.750	0.379	0.772	0.675
tai385	0.158	0.861	0.453	0.694	0.612
tai75a	0.156	0.715	**0.135**	0.661	0.599
tai75b	0.075	0.732	0.251	0.611	0.442
tai75c	0.258	0.682	**0.002**	0.032	0.032
tai75d	0.277	0.549	**0.036**	0.459	0.458
Avg.	0.198	0.697	0.253	0.541	0.467

Figure 3 shows two large trees, with labels drawn only for 1 in 4 solutions because the trees contain 170 solutions generated by memetic algorithms (2 worst solutions of each algorithm were omitted); the previous trees contain only 35 solutions each.

In each of the figures there is shown one example of a tree for highly clustered set of solutions and one example of a tree for solutions spread in the search space.

The tree generated for instance tai385 and distance d_e (Figure 2, left) shows very different solutions: the height of this tree is 0.263 and, what is more important, levels of links between solutions are also very high, the smallest being around 0.1. The shape of this tree might be called 'wide and high'. On the contrary, the tree for tai75d and d_{pn} (Figure 2, right) presents highly clustered solutions: although the height of this tree is 0.15, all solutions are in one of only 2 clusters of identical solutions (from the point of view of d_{pn}). The shape of this tree might be called 'narrow and with wide leaves'.

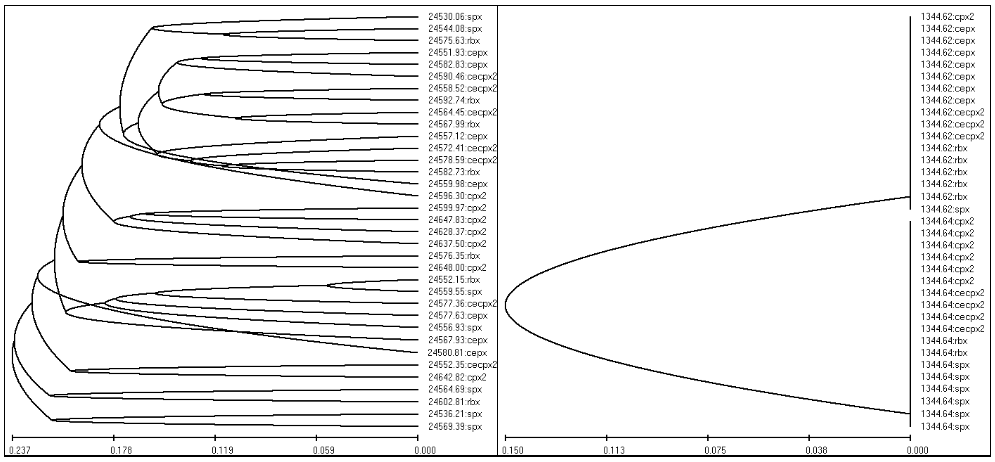

Figure 2. Sorted clustering trees for 7 best solutions of each of 5 memetic algorithms (35 solutions in total). Instance tai385, distance d_e (left) and tai75b, distance d_{pn} (right).

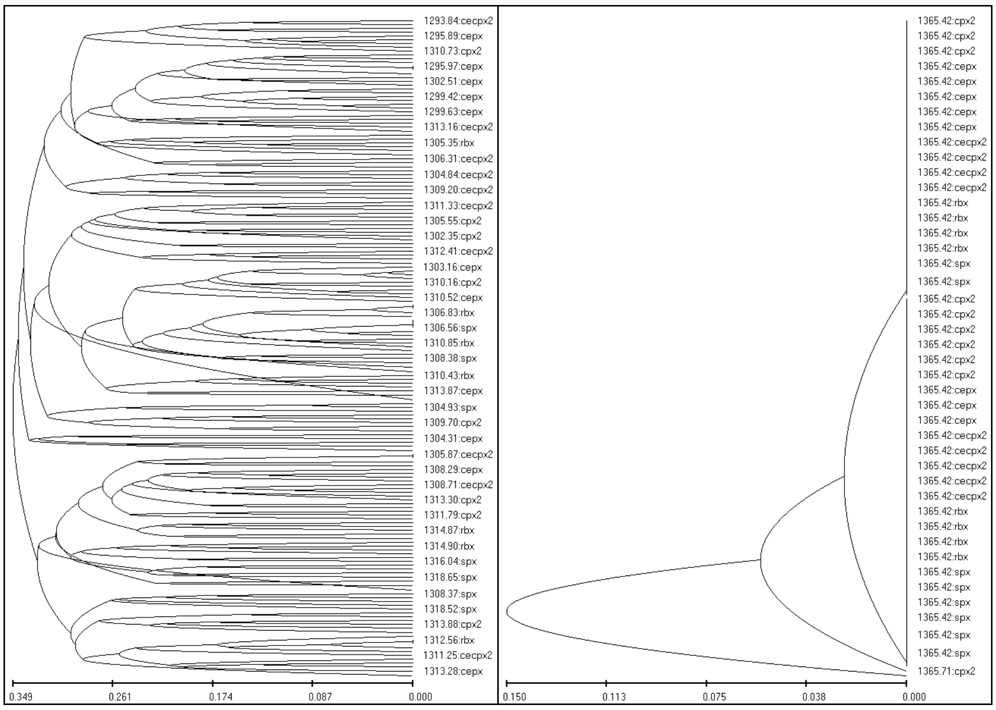

Figure 3. Sorted clustering trees for 34 solutions of each of 5 memetic algorithms (170 solutions in total). Instance c199, distance d_{pn} (left) and tai75d, distance d_e (right). Only 1 in 4 solutions is labelled.

Moreover, Figure 2 shows that solutions of different algorithms are mixed in the search space rather than clustered by the type of the algorithm (examine labels of linked leaves). This observation is independent on the hardness of an instance and was also made by the author in case of other instances, but the related trees are not shown here[3]. Nevertheless, this fact leads to the conclusion that all the memetic algorithms search in the same area of the solution space, independently on the crossover operator used.

Trees in Figure 3 confirm the observations made earlier: solutions of instance c199 are hardly clustered with respect do d_{pn} (see the height of the tree and individual links; also note the shape of this tree: 'wide and high'), whereas solutions of tai75d are highly clustered with respect to d_e, the tree is 'narrow and with wide leaves' and with two worst solutions being distant to all others.

7 Conclusions

The conducted experiment and analyses indicate that not all instances reveal 'narrow big valleys' which could, in turn, make the search by optimisation algorithms easy (if such algorithms preserved certain properties of solutions). The easy instances are: c100b, c120, c50, f134, f71, tai100b, tai75a, tai75c, tai75d, and perhaps tai75b (d_e) and tai150a (d_{pn}). The other instances reveal 'wider valleys', harder for optimisation.

Another conclusion is that the crossover operators do not significantly influence the part of search space the algorithm with certain crossover aims at, because solutions of different algorithms (crossover operators) are mixed with each other.

Further analyses also revealed that there is a relationship between quality of solutions generated by a certain algorithm and average distance between solutions of all algorithms. The author created rankings of instances with respect to average quality of solutions of each algorithm (based on columns 3, 5, 7, 9, 11 of Table 1) and also rankings of instances with respect to average distance d_e and d_{pn} of solutions. Values of the Kendall's tau coefficient for compared rankings (one algorithm vs. one distance) varied from 0.66 to 0.73, thus indicating high similarity of these rankings and tight relationship between average distance and quality.

Bibliography

[1] Boese, K. D. (1995): "Cost versus distance in the traveling salesman problem". Technical Report [TR-950018]. UCLA CS Department, California, USA.

[2] Clarke, G., and Wright, J. (1964): "Scheduling of vehicles from a central depot to a number of delivery points". In: *Operations Research*, 12, 568-582.

[3] Fisher, M. L. (1994): "Optimal solution of vehicle routing problems using minimum K-trees". In: *Operations Research*, 42, 626-642.

[4] Gillet, B. E., and Miller, L. R. (1974): "A heuristic algorithm for the vehicle dispatch problem". In: *Operations Research*, 22, 340-349.

[5] Jaszkiewicz, A., and Kominek, P. (2003): "Genetic local search with distance preserving recombination operator for a vehicle routing problem". In: *European Journal of Operational Research*, 151, 352-364.

[3] All trees generated for these solutions might be found at: http://www.cs.put.poznan.pl/mkubiak/ /research/cvrp

[6] Jaszkiewicz, A., Kominek, P., and Kubiak, M. (2004): "Adaptation of the genetic local search algorithm to a car sequencing problem". In: *Proceedings of the 2004 National Conference on Evolutionary Algorithms and Global Optimization*, Kazimierz Dolny, Poland, May 2004.

[7] Jones, T., and Forrest, S. (1995): "Fitness distance correlation as a measure of problem difficulty for genetic algorithms". *Santa Fe Institute Working Paper 95-02-022.*

[8] Jozefowiez, N., Semet, F., and Talbi, E. (2002): "Parallel and hybrid models for multi-objective optimization: application to the vehicle routing problem". In: *Proceedings of the Parallel Problem Solving from Nature*.

[9] Kubiak, M. (2004): "Systematic construction of recombination operators for the vehicle routing problem". In: *Foundations of Computing and Decision Sciences*, 29 (3), 205-226.

[10] Kubiak, M. (2005): "Distance metrics and fitness-distance analysis for the capacitated vehicle routing problem". In: *MIC2005, 6th Metaheuristics International Conference*, Vienna, Austria, July 2005.

[11] Merz, P. (2004): "Advanced fitness landscape analysis and the performance of memetic algorithms". In: *Evolutionary Computation*, 12 (3), 303-325.

[12] Merz, P., and Freisleben, B. (1999): "Fitness landscapes and memetic algorithms design". In: D. Corne, et al. (eds.), *New Ideas in Optimization*, McGraw-Hill.

[13] Merz, P., and Freisleben, B. (2000): "Fitness landscape analysis and memetic algorithms for the quadratic assignment problem". In: *IEEE Transactions on Evolutionary Computation*, 4 (4), 337-352.

[14] Potvin, J.-Y., and Bengio, S. (1996): "The vehicle routing problem with time windows part ii: genetic search". In: *INFORMS Journal of Computing*, 8 (2).

[15] Prins, C. (2001): "A simple and effective evolutionary algorithm for the vehicle routing problem. In: *MIC2001, 4th Metaheuristics International Conference*, July 2001.

[16] Rochat, Y., and Taillard, E. D. (1995): "Probabilistic diversification and intensification in local search for vehicle routing". In: *Journal of Heuristics*, 1, 147-167.

[17] Sneath, P. H. A., and Sokal, R. R. (1973): "Numerical Taxonomy". Freeman & Co., San Francisco.

[18] Tavares, J., et al. (2003): "On the influence of GVR in vehicle routing". *SAC'2003*, Melbourne, Florida, USA.

[19] Watson, J. P, et al. (1998): "The travelling salesrep problem, edge assembly crossover, and 2-opt". In: *Parallel Problem Solving from Nature (PPSN-V).*