An Adaptive Multi-Crossover Population Algorithm for Solving Routing Problems

E. Osaba, E. Onieva, R. Carballedo,
F. Diaz, and A. Perallos

Deusto Institute of Technology (DeustoTech), University of Deusto,
Av. Universidades 24, Bilbao 48007, Spain
[e.osaba, enrique.onieva, roberto.carballedo,fernando.diaz,
asier.perallos]@deusto.es

Abstract. Throughout the history, Genetic Algorithms (GA) have been widely applied to a broad range of combinatorial optimization problems. Its easy applicability to areas such as transport or industry has been one of the reasons for its great success. In this paper, we propose a new Adaptive Multi-Crossover Population Algorithm (AMCPA). This new technique changes the philosophy of the basic genetic algorithms, giving priority to the mutation phase and providing dynamism to the crossover probability. To prevent the premature convergence, in the proposed AMCPA, the crossover probability begins with a low value, and varies depending on two factors: the algorithm performance on recent generations and the current generation number. Apart from this, as another mechanism to avoid premature convergence, our AMCPA has different crossover functions, which are used alternatively. We test the quality of our new technique applying it to three routing problems: the Traveling Salesman Problem (TSP), the Capacitated Vehicle Routing Problem (CVRP) and the Vehicle Routing Problem with Backhauls (VRPB). We compare the results with the ones obtained by a basic GA to conclude that our new proposal outperforms it.

Keywords: Adaptive Population Algorithm, Genetic Algorithm, Routing Problems, Combinatorial Optimization, Intelligent Transport Systems

1 Introduction

Since its proposal in the '70s, genetic algorithm (GA) has become one of the most successful meta-heuristic techniques for solving combinatorial optimization problems. GAs are based on the genetic process of living organisms and in the law of the species evolution, proposed by Darwin. The basic principles of this technique were proposed by Holland [1], trying to imitate the natural selection process and the strongest specimens survival. Even though, its practical use for solving complex problems was shown later by De Jong [2] and Goldberg [3]. From that moment, GAs has been the focus of a large number of papers and

books [4, 5], and they have been applied in a wide range of fields, like transport [6], software engineering [7] or industry [8].

In this paper, we present an Adaptive Multi-Crossover Population Algorithm (AMCPA) for solving routing problems. This new meta-heuristic is a variant of the basic GA. It prioritizes the local optimization (mutation), applying crossover operators only when they would be beneficial to the search process. In our AMCPA the crossover probability varies, depending on the search performance on recent generations and the current generation number. This dynamism helps our technique to prevent premature convergence. Apart from this, the proposed AMCPA has multiple crossover functions, which are applied alternatively.

Adjusting the control parameters of the GAs has always been one of the most controversial questions in the field of genetic algorithms. Related works have been done since the 80's [9] until today [10]. Concretely, the idea of adapting crossover and mutation probabilities (p_c and p_m) to improve the performance of GAs has been studied since long time ago, for example in [11] and [12], but it is also subject of many studies nowadays. Below were mentioned several examples of works on this topic, being the whole literature for this field much larger. In [13], for example, a genetic algorithm that adapts its p_c and p_m in function of the population fitness difference and the maximum fitness value is presented. In [14] and [15], a GA that uses fuzzy logic to adaptively tune p_c and p_m is introduced. In these proposals, a clustering technique is used to split the population in clusters. Then, a fuzzy system determines the p_c and p_m depending on the best and worst chromosome of each cluster. In [16] is proposed a GA that, besides adapting the p_m, determines the types of replacing genes in the mutation procedure. In [17] some improvements on adaptive GAs for reliability-related applications are introduced. In that work, the authors present a simple parameter-adjusting method, using the fitness average and variance of the population. In [18] an adaptive algorithm for optimizing the design of high pressure hydrogen storage vessel is presented. That algorithm adapts p_c and p_m depending on the fitness value of each individual. Finally, another example of adapting p_c and p_m is the one presented in [19]. In that work an improved adaptive genetic algorithm based on hormone modulation mechanism to solve the job-shop scheduling problem is proposed.

Regarding the multi-crossover, this mechanism has been used less than the previously explained one. Anyway, it has also been studied before, long time ago and nowadays. In [20], for example, an adapting crossover technique for an population algorithm is presented, which varies the crossover operator and its utilization frequency. The algorithm proposed in that work uses two crossovers functions depending on the population situation in the solution space. Another example is [21]. In this work a strategy adaptive genetic algorithm is proposed for solving the well-known Traveling Salesman Problem (TSP) [22]. This algorithm works with three different crossover functions. The choice of the function is decided partly by the quality of each of them and partly at random.

After a brief analysis of the state of the art, we detail the most innovative aspects of the new technique we propose:

– Our AMCPA reverses the philosophy of conventional GAs. It starts with a high values of p_m and a very low or null value for p_c. This fact is based on our previous work [23].
– Our proposal adapts its p_c depending on the current generation number and the search performance in recent iterations, instead of relying on the population fitness, as most previous studies.
– The proposed algorithm combines the p_c adaptation and the multi-crossover mechanism, something that has not been done frequently before.
– The introduced AMCPA is tested with routing problems. Traditionally, adaptive algorithm has not been applied to this type of problem.

This paper is structured as follows. In the following section we introduce our proposed technique. Then, we will show the results of our AMCPA applied to three different well known routing problems: TSP, Capacitated Vehicle Routing Problem (CVRP) [24] and Vehicle Routing Problem with Backhauls (VRPB) [25]. In the same section we compare the results obtained by our algorithm with the results of a basic GA. We finish this work with the conclusions and future work.

2 Our Adaptive Multi-Crossover Population Algorithm

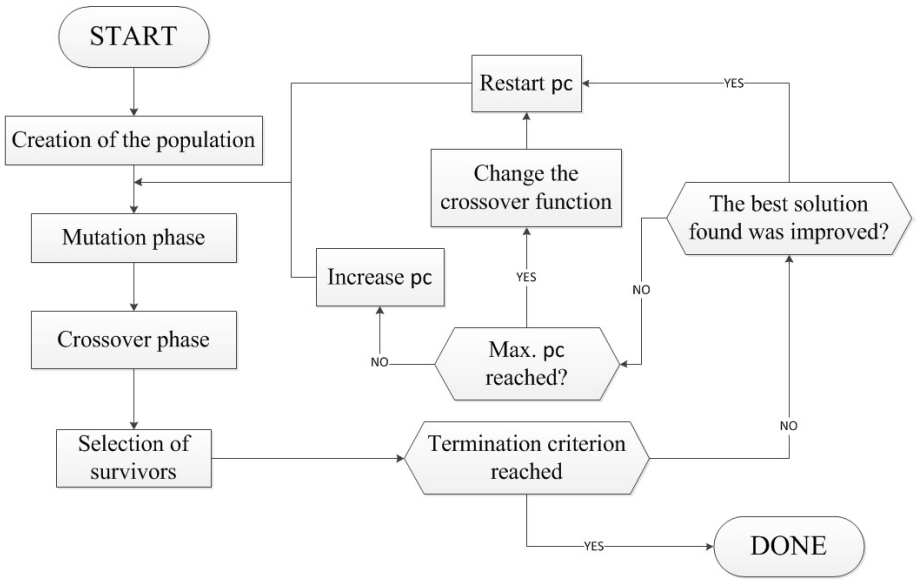

Fig. 1. Flowchart of the algorithm

As already mentioned, our AMCPA is a variant of a conventional GA. Its flowchart can be seen in figure 1. The proposed technique reverses the philosophy

of conventional GAs, giving higher priority to the individual optimization, provided by the mutation phase, and giving less importance to the crossovers phase. These fundamentals are based on our recently published study [23], in which we analyze the blind crossover suitability in GAs solving routing problems. In that work we check our theory, which stands that the crossover phase is not efficient for the optimization capacity of the technique when it is applied to routing problems using path encoding. For this reason, the proposed AMCPA offers a greater role to the mutation phase. Despite this, we consider that the crossovers between different individuals can be beneficial to maintain the population diversity. Therefore, in the proposed AMCPA we try to fit the p_c to the search process needs. Apart from that, as an additional tool to avoid the premature convergence, our AMCPA has a multi-crossover mechanism, which changes the crossover operator of the technique for all the population. These changes are made based on various concepts which will be explained later. Below, we will describe these mechanisms.

2.1 Adaptive Mechanism

Regarding the p_m, in our AMCPA, all individuals in the population go through the mutation process every generation. This would be equivalent to having a p_m equals to 1.0. On the other hand, in the proposed method the p_c starts with a very low value, close to 0.0. The latter parameter is modified as search progresses, increasing or restarting its value to 0. The modification is performed based on the improvement in the best solution found in the last generation. This modification is based on the following criteria:

- *The best solution found by the technique has been improved in the last generation*: This means that the search process evolves correctly and that it is not necessary to diversify the population. In this case, the value of p_c is restarted.
- *The best solution found by the technique has not been improved in the last generation*: In this case, it could be considered that the search is in a bump. This means that the search process could be trapped in a local optimum, or that the population could be concentrated in the same region of the solution space. At this time, increasing the population diversification using crossover operators would be beneficial. With this intention p_c is increased.

Whenever the best solution found has not been improved over the previous generation, p_c increases based on the following function, where N represents the number of generations executed without improvements, NG the total number of generations executed and NMF represents the size of the mutation operator neighborhood:

$$p_c = p_c + \frac{N^2}{NMF^2} + \frac{NG}{NMF^2}$$

As seen in the formula above, p_c increases proportionally to the total number of generations (NG) and the number of generations without any improvement in the best solution (N).

2.2 Multi-Crossover mechanism

In relation to the multi-crossover feature, as we have already said, our AMCPA has more than one crossover operator which are alternated during the execution. At the beginning, one operator is assigned at random. Along the execution, this function will be randomly replaced by another available, allowing repetitions. For this purpose, a maximum p_c value is defined. If over the generations the p_c value exceeds that maximum, the crossover function will be replaced at random by another one, and p_c will be restarted with the initial value.

The maximum p_c value is an adjustable parameter, which has to be high enough to prevent a premature function change. Furthermore, to avoid an excessive runtime waste, the value cannot be too high.

This mechanism allows a diversification of the population much more efficient than other similar techniques. That is, prevent the algorithm from being trapped in a local optimum.

3 Experimentation

In this section we show in detail the results of applying our AMCPA three well-known combinatorial optimization problems. As we have mentioned, the technique proposed in this paper is a basic GA variation. For that reason, we compare the results obtained by a traditional GA, and our new AMCPA. For both algorithms we have used similar functions and parameters, so that the only difference between them is their working way. This method of comparing meta-heuristics is the most reliable way to determine which technique gets better results. The tests were performed with the three different problems that have been mentioned in the introduction: TSP, CVRP and VRPB. All these problems are well-known in combinatorial optimization and they are used in many studies annually [26–33]

3.1 Parameters of the algorithms

For both algorithms, the population is composed by 50 individuals, which are created randomly. The aim of this study is to make a comparison between our AMCPA and a GA, for that reason the population size is not very important, as long as the two meta-heuristics have the same. Regarding the selection and survivor phases, same function is used for both in all instances, which is the 0.5 elitist - 0.5 random. About the ending criteria, the execution of both algorithms finishes when there are a generation number proportional to the size of the neighborhood (obtained by the mutation operator) without improvements in the best solution found. The individuals encoding mode is the *Path Encoding*.

For the GA, the p_m is 0.05 while the p_c is 0.95. In the case of the proposed AMCPA, the p_c starts at 0.0. When the best solution found is not improved, the p_c increases following the formula shown in 2.1, otherwise, it returns to 0.0.

For the TSP, the crossover functions used for our AMCPA are Order Crossover (OX) [34], Modified Order Crossover (MOX) [35] and Order Based

Crossover (OBX) [36]. These functions have been widely used since their creation [37–41]. On the other hand, OX is used as crossover function for the GA. The mutation function for both techniques is the 2-opt [42], which has been very used since its formulation [43, 44].

For the CVRP and VRPB, the crossover functions used for the proposed AMCPA are the *Half Crossover* (HX) and *Half Random Crossover* (HRX). These functions are a particular case of the traditional crossover, in which the cut point is made always in the middle of the path. With HX, first, the 50% of the best routes in one randomly chosen parent are selected and inserted in the child. Then, the nodes already inserted are removed from the other parent. Finally, the remaining nodes are inserted in the same order in the final solution, creating new routes. The HRX working way is similar to HX. In this case, in the first step, the routes selected from one of the parents are chosen randomly, instead of selecting the best ones. For the GA the crossover function used is the HX.

Continuing with the CVRP and VRPB, regarding the mutation function, we have used for both techniques the called *Vertex Insertion Routes*. This function selects and extracts one random node from a random route. Then, the node is re-inserted in a random position in another randomly selected route. New routes creation is possible with this function.

3.2 Results

All the tests were performed on an Intel Core i5 2410 laptop, with 2.30 GHz and a 4 GB of RAM. For each run we display the total average, the best result obtained and the standard deviation. The objetive function used in the three problems is the total traveled distance. We also show the average runtime, in seconds. In order to determine if GB average is significantly different than the averages obtained by GA, we perform Students t-test. The t statistic has the following form [45]:

$$t = \frac{\overline{X_1} - \overline{X_2}}{\sqrt{\frac{(n_1-1)SD_1^2+(n_2-1)SD_2^2}{n_1n_2-2}}\frac{n_1+n_2}{n_1n_2}}$$

where:

$\overline{X_1}$: Average of our AMCPA,
SD_1: Standar deviation of our AMCPA,
$\overline{X_2}$: Average of GA,
SD_2: Standar deviation of GA,
n_1: Our AMCPA size,
n_2: GA size,

The t values shown can be positive, neutral, or negative. The positive value of t indicates that our proposal is significantly better than GA. In the opposite

case, GA obtains better solutions. If t is neutral, the difference between the two algorithms is not significant. We stated confidence interval at the 95% confidence level ($t_{0.05} = 2.021$).

Each experiment is repeated 20 times. Instances for the TSP were obtained from the TSPLIB Benchmark [46]. For the CVRP, the instances were picked from the CVRP set of Christofides and Eilon (*http://neo.lcc.uma.es/vrp* [1]). The name of each TSP and CVRP instances has a number that displays the number of nodes it has. Tables 1 and 2 show the results for these problems.

Instance		Proposed AMCPA				Genetic Algorithm				t test
Name	Optima	Avg.	S. dev.	Best	Time	Avg.	S. dev.	Best	Time	t
Oliver30	420	**427.5**	4.4	420	0.08	435.3 (+1.82)	15.3	420	0.19	+
Eilon50	425	**440.5**	6.6	430	0.42	469.9	17.5	435	1.59	+
Eil51	426	**445.0**	5.9	441	0.36	465.7	10.5	441	1.33	+
Berlin52	7542	**7805.2**	284.7	7542	0.29	8040.1	188.4	7745	1.36	+
St70	675	**706.5**	16.1	692	0.81	750.2	30.1	707	4.18	+
Eilon75	535	**575.0**	9.5	547	1.27	615.4	14.7	585	6.22	+
Eil76	538	**578.1**	12.2	566	1.28	610.6	12.2	558	6.96	+
KroA100	21282	**22125.3**	460.3	21608	2.25	22270.4	711.0	21566	14.84	+
KroB100	22140	**23043.7**	355.6	22536	2.28	23565.4	489.3	23253	13.54	+
KroC100	20749	**21550.8**	355.6	20785	2.40	22572.2	713.6	22271	16.52	+
KroD100	21294	**22125.5**	457.0	21725	2.25	23246.8	424.7	22162	12.92	+
KroE100	22068	**23196.7**	484.9	22611	2.54	23329.6	712.4	22412	12.72	+
Eil101	629	**678.1**	13.6	657	4.11	725.8	22.2	696	18.38	+
Pr107	44303	**45361.2**	953.1	44438	4.50	46742.2	1404.7	45833	18.21	+
Pr124	59030	**60578.6**	752.4	59030	6.86	62203.0	1223.3	60127	22.34	+
Pr136	96772	**101712.4**	1548.1	98125	7.42	104308.7	2425.1	99835	46.79	+
Pr144	58537	**60259.8**	1128.6	59061	9.57	62892.2	2552.9	60275	50.12	+
Pr152	73682	**76225.4**	1138.0	74518	10.32	77925.1	2862.3	74250	57.25	+

Table 1. Results of our AMCPA and GA for the TSP

For the VRPB we have used 10 instances. The first 6 were obtained from the VRPTW Benchmark of Solomon (*http://neo.lcc.uma.es/vrp*). In this case, the time constraints have been removed, but vehicle capacities and the amount of customer demands are retained. Apart from this, we also have been modified the demands nature with the aim of creating pickup and deliveries. The remaining 4 instances were obtained from the CVRP set of Christofides and Eilon. In these instances, the vehicle capacities and the number of nodes have been maintained, but the demand types have been also changed to have pickups and deliveries. For these cases the optimums are not shown, since they are not typical VRPB instances, therefore, these values are unknown. The last table (table 3) shows the results for this problem.

[1] Last update: January 2013

Instance		Proposed AMCPA				Genetic Algorithm				t test
Name	Optima	Avg.	S. dev.	Best	Time	Avg.	S. dev.	Best	Time	t
En22k4	375	395.6	7.1	375	2.08	**388.4**	15.1	375	3.78	-
En23k3	569	**611.8**	43.3	569	2.24	646.5	38.6	592	3.78	+
En30k3	534	**560.7**	25.8	534	3.02	570.8	25.1	535	6.74	+
En33k4	835	**903.5**	20.2	869	3.15	921.1	27.2	882	7.40	+
En51k5	521	**617.6**	27.2	587	4.56	680.8	47.1	604	18.17	+
En76k7	682	**813.0**	62.8	762	10.17	878.9	44.8	793	57.17	+
En76k8	735	**876.5**	32.1	819	10.67	953.4	46.5	920	53.86	+
En76k10	830	**965.8**	17.6	921	11.04	1029.6	34.4	956	55.27	+
En76k14	1021	**1170.7**	46.3	1135	7.39	1191.6	35.0	1125	80.54	+
En101k8	815	**1012.0**	59.6	916	17.95	1081.0	43.3	1011	100.25	+
En101k14	1071	**1272.6**	47.1	1201	18.60	1369.7	49.8	1308	120.84	+

Table 2. Results of our AMCPA and GA for the CVRP

Instance	Proposed AMCPA				Genetic Algorithm				t test
Name	Avg.	S. dev.	Best	Time	Avg.	S. dev.	Best	Time	t
C101	724.4	41.9	627	8.25	**723.0**	71.9	624	34.93	*
C201	**652.4**	12.4	617	4.04	849.9	98.8	744	25.43	+
R101	**962.2**	38.7	875	5.86	1081.2	70.7	957	29.51	+
R201	**1105.2**	41.7	1021	12.54	1335.7	112.4	1224	56.94	+
RC101	**595.2**	47.1	529	2.13	659.4	64.2	563	5.45	+
RC201	**1221.6**	90.8	1167	18.35	1505.3	92.2	1367	57.59	+
En30k4	**534.5**	27.8	500	1.57	583.4	67.0	520	3.53	+
En33k4	**812.9**	45.8	787	1.85	848.9	44.4	751	4.64	+
En51k5	**688.1**	34.8	636	4.01	727.6	30.6	680	9.41	+
En76k8	**912.8**	43.5	798	7.55	1008.8	40.5	927	28.61	+

Table 3. Results of our AMCPA and GA for the VRPB

3.3 Analysis of results

Viewing the results obtained the conclusion that can be drawn is clear. The proposed technique outperforms the GA in terms of solution quality and runtimes. The reason why our algorithm needs lower runtime is logical. If mutation and crossover functions are compared, the last ones needs more time to execute, since they operate with two different solutions, and their working way is more complex that the mutation. On the other hand, the mutation operates with one solution and it is a simple modification in a chromosome which can be made in a minimum time. Our AMCPA makes fewer crossovers than the GA. This fact is perfectly reflected in the runtimes, giving a great advantage to our technique.

The reason why the proposed AMCPA gets better results can also be explained, and it is based on the conclusions obtained in our recent study [23]. Crossovers between different individuals are very useful resources if we want to make jumps in the solution space. Using crossovers helps a broad exploration of the solution space, but does not help to make an exhaustive search. To get

a deeper search, the existence of a function that takes care of optimizing the solutions independently becomes necessary. The mutation function can handle this goal easily.

With all this, our AMCPA is a technique that is able to perform a thorough and intense search in promising regions of the solution space using the mutation function. While it do this, it uses the crossover function in case the search is in a bump, in order to avoid local optimums. Using the crossovers, the current population is expanded through the entire solution space, and will be easier to find regions that allow the search to reach better results. This diversification is enhanced thanks to the multi-crossover, allowing a broader exploration.

By contrast, with the GA basic structure, the search performed by the algorithm comprises a large percentage of the solution space, but has a smaller capacity to deepen in those areas which are most promising. This means that, finally, the GA obtains worse results than the DEA.

4 Conclusions and further work

In this paper we have presented an Adaptive Multi-Crossover Population Algorithm for solving routing problems, which is a variation of the conventional genetic algorithm. Our AMCPA reverses GAs conventional philosophy, giving priority to the individual autonomous improvement, making crossovers only when they are beneficial for the search process. The proposed technique has two mechanisms to avoid the premature convergence, helping to the population diversity. These mechanisms are the crossover probability adaption and the use of multiple crossover operators.

Initially we have introduced our new meta-heuristic, explaining how it works. Then, we have shown the results obtained by applying it to three different routing problems. We have compared these outcomes with obtained by a basic GA, to conclude that our method gets better results. Finally, we have reasoned why our new technique is better than the GA.

As future work, we will compare the performance of our technique with other approaches of similar philosophy that we can find in the literature. In addition, we are planning to apply our new proposal to real life routing problems. At this time, we are planning its application to a dynamic distribution system of car windscreen repairs. In this case the problem is designed as a dynamic CVRP, wherein the routes may be re-planned according to the needs of the customers. Apart from this, we are planning to extend our technique, turning it into an island-based meta-heuristic. This new technique will use different crossover functions for the different populations evolving in each island, and will make transfers of individuals between them.

References

1. Holland, J.H.: Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press (1992)

2. De Jong, K.: Analysis of the behavior of a class of genetic adaptive systems. PhD thesis, University of Michigan, Michigan, USA (1975)
3. Goldberg, D.: Genetic algorithms in search, optimization, and machine learning. Addison-Wesley Professional (1989)
4. Harvey, I.: The microbial genetic algorithm. Advances in Artificial Life. Darwin Meets von Neumann **5778** (2011) 126–133
5. Affenzeller, M., Wagner, S., Winkler, S.: Genetic algorithms and genetic programming: modern concepts and practical applications. Volume 6. Chapman & Hall/CRC (2009)
6. Moon, I., Lee, J.H., Seong, J.: Vehicle routing problem with time windows considering overtime and outsourcing vehicles. Expert Systems with Applications **39**(18) (2012) 13202–13213
7. Martínez-Torres, M.: A genetic search of patterns of behaviour in oss communities. Expert Systems with Applications **39**(18) (2012) 13182–13192
8. Gao, J., Gen, M., Sun, L., Zhao, X.: A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems. Computers & Industrial Engineering **53**(1) (2007) 149–162
9. Grefenstette, J.J.: Optimization of control parameters for genetic algorithms. Systems, Man and Cybernetics, IEEE Transactions on **16**(1) (1986) 122–128
10. Fernandez-Prieto, J., Gadeo-Martos, M., Velasco, J.R., et al.: Optimisation of control parameters for genetic algorithms to test computer networks under realistic traffic loads. Applied Soft Computing **11**(4) (2011) 3744–3752
11. Schaffer, J.D., Morishima, A.: An adaptive crossover distribution mechanism for genetic algorithms. In: Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application, L. Erlbaum Associates Inc. (1987) 36–40
12. Davis, L.: Adapting operator probabilities in genetic algorithms. In: Proceeding of the Third International Conference on Genetic Algorithms. (1989) 61–69
13. Srinivas, M., Patnaik, L.M.: Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Transactions on Systems, Man and Cybernetics **24**(4) (1994) 656–667
14. Zhang, J., Chung, H.S., Zhong, J.: Adaptive crossover and mutation in genetic algorithms based on clustering technique. In: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, GECCO 2005, ACM (2005) 1577–1578
15. Zhang, J., Chung, H.S., Lo, W.L.: Clustering-based adaptive crossover and mutation probabilities for genetic algorithms. IEEE Transactions on Evolutionary Computation **11**(3) (2007) 326–335
16. Vafaee, F., Nelson, P.C.: A genetic algorithm that incorporates an adaptive mutation based on an evolutionary model. In: Proceedings of the 2009 International Conference on Machine Learning and Applications, IEEE (2009) 101–107
17. Ye, Z., Li, Z., Xie, M.: Some improvements on adaptive genetic algorithms for reliability-related applications. Reliability Engineering & System Safety **95**(2) (2010) 120–126
18. Xu, P., Zheng, J., Chen, H., Liu, P.: Optimal design of high pressure hydrogen storage vessel using an adaptive genetic algorithm. International Journal of Hydrogen Energy **35**(7) (2010) 2840–2846
19. Wang, L., Tang, D.b.: An improved adaptive genetic algorithm based on hormone modulation mechanism for job-shop scheduling problem. Expert Systems with Applications **38**(6) (2011) 7243–7250
20. Spears, W.M.: Adapting crossover in evolutionary algorithms. In: Proceedings of the 1995 Conference on Evolutionary Programming. (1995) 367–384

21. Mukherjee, S., Ganguly, S., Das, S.: A strategy adaptive genetic algorithm for solving the travelling salesman problem. In: Swarm, Evolutionary, and Memetic Computing. Springer (2012) 778–784
22. Lawler, E., Lenstra, J., Kan, A., Shmoys, D.: The traveling salesman problem: a guided tour of combinatorial optimization. Volume 3. Wiley New York (1985)
23. Osaba, E., Carballedo, R., Diaz, F., Perallos, A.: Analysis of the suitability of using blind crossover operators in genetic algorithms for solving routing problems. In: Proceedings of the IEEE 8th International Symposium on Applied Computational Intelligence and Informatics. IEEE (2013) 17–23
24. Laporte, G.: The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operational Research **59**(3) (1992) 345–358
25. Golden, B., Baker, E., Alfaro, J., Schaffer, J.: The vehicle routing problem with backhauling: two approaches. In: Proceedings of the Twenty-first Annual Meeting of SE TIMS, South Carolina USA (1985) 90–92
26. Liefooghe, A., Humeau, J., Mesmoudi, S., Jourdan, L., Talbi, E.: On dominance-based multiobjective local search: design, implementation and experimental analysis on scheduling and traveling salesman problems. Journal of Heuristics **18**(2) (2012) 317–352
27. Bae, J., Rathinam, S.: Approximation algorithms for multiple terminal, hamiltonian path problems. Optimization Letters **6**(1) (2012) 69–85
28. Sarin, S.C., Sherali, H.D., Yao, L.: New formulation for the high multiplicity asymmetric traveling salesman problem with application to the chesapeake problem. Optimization Letters **5**(2) (2011) 259–272
29. Li, W., Shi, Y.: On the maximum tsp with γ-parameterized triangle inequality. Optimization Letters **6**(3) (2012) 415–420
30. Mattos Ribeiro, G., Laporte, G.: An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. Computers & Operations Research **39**(3) (2012) 728–735
31. Ngueveu, S., Prins, C., Wolfler Calvo, R.: An effective memetic algorithm for the cumulative capacitated vehicle routing problem. Computers & Operations Research **37**(11) (2010) 1877–1885
32. Gajpal, Y., Abad, P.: Multi-ant colony system (macs) for a vehicle routing problem with backhauls. European Journal of Operational Research **196**(1) (2009) 102–117
33. Anbuudayasankar, S., Ganesh, K., Lenny Koh, S., Ducq, Y.: Modified savings heuristics and genetic algorithm for bi-objective vehicle routing problem with forced backhauls. Expert Systems with Applications **39**(3) (2012) 2296–2305
34. Davis, L.: Applying adaptive algorithms to epistatic domains. In: Proceedings of the international joint conference on artificial intelligence. Volume 1. (1985) 161–163
35. Ray, S., Bandyopadhyay, S., Pal, S.: New operators of genetic algorithms for traveling salesman problem. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. Volume 2., IEEE (2004) 497–500
36. Syswerda, G.: Schedule optimization using genetic algorithms. Handbook of genetic algorithms (1991) 332–349
37. Sharma, S., Gupta, K.: Solving the traveling salesmen problem through genetic algorithm with new variation order crossover. In: Emerging Trends in Networks and Computer Communications (ETNCC), 2011 International Conference on, IEEE (2011) 274–276
38. Prins, C.: A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & Operations Research **31**(12) (2004) 1985–2002

39. Albayrak, M., Allahverdi, N.: Development a new mutation operator to solve the traveling salesman problem by aid of genetic algorithms. Expert Systems with Applications **38**(3) (2011) 1313–1320
40. Rocha, M., Sousa, P., Cortez, P., Rio, M.: Quality of service constrained routing optimization using evolutionary computation. Applied Soft Computing **11**(1) (2011) 356–364
41. Wang, C., Zhang, J., Yang, J., Hu, C., Liu, J.: A modified particle swarm optimization algorithm and its application for solving traveling salesman problem. In: Proceedings of the 2005 International Conference on Neural Networks and Brain. Volume 2., IEEE (2005) 689–694
42. Lin, S.: Computer solutions of the traveling salesman problem. Bell System Technical Journal **44**(10) (1965) 2245–2269
43. Tarantilis, C., Kiranoudis, C.: A flexible adaptive memory-based algorithm for real-life transportation operations: Two case studies from dairy and construction sector. European Journal of Operational Research **179**(3) (2007) 806–822
44. Bianchessi, N., Righini, G.: Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. Computers & Operations Research **34**(2) (2007) 578–594
45. Nikolić, M., Teodorović, D.: Empirical study of the bee colony optimization (bco) algorithm. Expert Systems with Applications **40**(1) (2013) 4609–4620
46. Reinelt, G.: Tsplib: A traveling salesman problem library. ORSA journal on computing **3**(4) (1991) 376–384