AMCPA: A Population Metaheuristic With Adaptive Crossover Probability and Multi-Crossover Mechanism for Solving Combinatorial Optimization Problems

Eneko Osaba[1], **Fernando Diaz**[2], **Enrique Onieva**[1], **Roberto Carballedo**[1], **and Asier Perallos**[1]

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

[2]Telecommunications Department, University of Deusto
Av. Universidades 24, Bilbao 48007, Spain
fernando.diaz@deusto.es

ABSTRACT

Combinatorial optimization is a field that receives much attention in artificial intelligence. Many problems of this type can be found in the literature, and a large number of techniques have been developed to be applied to them. Nowadays, population algorithms have become one of the most successful metaheuristics for solving this kind of problems. Among population techniques, Genetic Algorithms (GA) have received most attention due to its robustness and easy applicability. In this paper, an Adaptive Multi-Crossover Population Algorithm (AMCPA) is proposed, which is a variant of the classic GA. The presented AMCPA changes the philosophy of the basic GAs, 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, which is adapted every generation. Apart from this, as another mechanism to avoid premature convergence, different crossover functions are used alternatively. In order to prove the quality of the proposed technique, it is applied to six different combinatorial optimization problems, and its results are compared with the ones obtained by a classic GA. Additionally, the convergence behaviour of both techniques are also compared. Furthermore, with the objective of performing a rigorous comparison, a statistical study is conducted to compare these outcomes. The problems used during the test are: Symmetric and Asymmetric Traveling Salesman Problem, Capacitated Vehicle Routing Problem, Vehicle Routing Problem with Backhauls, N-Queens, and the one-dimensional Bin Packing Problem.

Keywords: Adaptive Evolutionary Algorithm, Metaheuristic, Multi-crossover, Genetic Algorithm, Combinatorial Optimization.

2010 Mathematics Subject Classification: 90C27, 90C59, 68T20.
1998 Computing Classification System: I.2.8., G.1.6.

1 Introduction

Nowadays, combinatorial optimization is one of the most studied fields in artificial intelligence. Problems arising in this area are the focus of many research studies every year (**?**). Some examples of this kind of problems are the Job-shop Scheduling Problem (**?**), and the Traveling Salesman Problem (TSP) (**?**). The main interest of these problems lies on their applicability to real life and their complexity. Being NP-Hard (**?**), a large number of techniques have been developed throughout the history with the aim of being applied in this field. Some of the most commonly used techniques are the tabu search (**?**), and the simulated annealing (**?**).

Additionally, population based algorithms have become one of the most successful approaches for solving combinatorial optimization problems. As is well known, these type of techniques work with one (or more than one) population of solutions, which evolves along the algorithm execution. Thanks to their robustness and their adaptability to a wide variety of problems, many population based metaheuristics have been introduced along the history, as the genetic algorithm (GA) (**?**), distributed population algorithm (**?**; **?**), particle swarm optimization (**?**; **?**; **?**), cultural algorithm (**?**; **?**), and the ant colony system (**?**; **?**). Furthermore, in recent years some sophisticated population techniques have been proposed, such as the imperialist competitive algorithm (**?**; **?**), artificial bee colony (**?**; **?**), firefly algorithm (**?**; **?**), gravitationl search algorithm (**?**; **?**; **?**), or golden ball metaheuristic (**?**; **?**). Anyway, among all the population techniques, the GA is the one which has received most attention.

GAs was proposed in the '70s, in an attempt to imitate the genetic process of living organisms and the law of the evolution of species. The basic principles of the GA were proposed by Holland in 1975 (**?**), even though its practical use for solving complex problems was demonstrated later by De Jong (**?**) and Goldberg (**?**). Thereafter, GAs has been the focus of a large number of research studies (**?**), and they have been applied in a wide range of fields, as industry (**?**) or transport (**?**; **?**).

The parameter adjustment is one of the most controversial questions in the field of GAs. Related studies have been done since the 80's (**?**), until today (**?**). Concretely, the idea of adapting crossover and mutation probabilities (p_c and p_m) in order to improve the GAs performance has been studied since long time ago (**?**; **?**). Anyway, it is still subject of many studies nowadays. The whole literature for this field is very large. Several examples are mentioned in this paper. In (**?**), a GA that adapts its p_c and p_m in function of the population fitness difference and the maximum fitness value is proposed. In (**?**; **?**), a GA that uses fuzzy logic to adaptively tune p_c and p_m is introduced. In these papers, 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 (**?**) a GA is proposed which adapts the p_m, and determines the types of replacing genes in the mutation procedure. In (**?**) some improvements on adaptive GAs for reliability-related applications are introduced. In that study, a simple parameter-adjusting method is presented, which uses the fitness average and variance of the population. In (**?**) an adaptive algorithm for optimizing the design of high pressure hydrogen storage vessel is presented. That technique adjusts p_m and p_c depending on the fitness value of each individual. Another example of adapting p_c and p_m is the one presented in (**?**). In that work an improved adaptive genetic algorithm based on hormone modulation mechanism to solve the job-shop scheduling problem is proposed.

In the following lines, some recent studies published last year (2013) in this field are introduced. The existence of these works demonstrates that the parameters adjustment in GAs is a topic of interest in the scientific community nowadays. In (**?**) an adaptive GA for large-size open stack problems is presented. This genetic approach combines a classical GA with and adaptive search strategy. That strategy uses a composite and dynamic fitness function which modifies the search landscape. In (**?**) an adaptive GA for daily optimal operation of cascade reservoirs is proposed. That adaptive GA adjusts the p_c and the p_m in order to improve the convergence speed of the GA. In (**?**) an adaptive GA for the time dependent inventory routing problems is introduced. That GA modifies the settings of the genetic parameter values according to the performance of the genetic operators. In that approach the fitness values of parents and offsprings are compared every generation. If the GA generates better offsprings during the genetic search process, p_c and p_m are increased, and vice versa. Finally, in (**?**; **?**) can be seen some other recent examples of adaptive GAs.

In regard to the multi-crossover mechanism, it has also been studied in previous studies. However, it has been used less than parameter adjustment methods. In (**?**), for example, an adapting crossover mechanism for a population algorithm is presented, which changes the crossover operator and the p_c. The algorithm proposed in that work uses two crossovers functions, which alternate depending on the situation of the population in the solution space. Another example can be found in (**?**). In that paper an adaptive GA is proposed for solving the well-known TSP. That GA works with three different crossover functions. The choice of the operatod is decided partly by the quality of each of them and partly at random. In the literature can be found an alternative approach for the multi-crossover. In this case, the multi-crossover mechanism is implemented by developing crossover functions which use more than two chromosomes (**?**). Although this approach is worth mentioning, it clearly falls outside the scope of the present work.

In this paper a new Adaptive Multi-Crossover Population Algorithm (AMCPA) for solving combinatorial optimization problems is presented. This new technique is a variant of the classic GA. In contrast to the classical philosophy of the GA, the introduced metaheuristic prioritizes the local improvement of the individuals (mutation), applying crossover operators only when they could be beneficial to the search process. In this way, in the presented AMCPA the crossover probability is adapted depending on the search performance on recent generations, and the current generation number. This dynamism helps the technique to prevent premature convergence. Besides this, the presented AMCPA uses multiple crossover functions, which are applied alternatively.

In order to prove the quality of the proposed technique, it has been applied to six different combinatorial optimization problems. The results obtained by our AMCPA in these six problems are compared with the ones obtained by a classic GA. Additionally, the convergence behaviour of both techniques are also analyzed and compared. Furthermore, with the objective of performing a rigorous comparison, a statistical study is conducted to compare these outcomes, performing the well-known normal distribution z-test. The main innovative aspects of the proposed AMCPA are the following:

1. The proposed AMCPA reverses the philosophy of conventional GAs. It starts with a very low or null value for p_c, a high values of p_m.
2. The introduced approach combines the multi-crossover mechanism and the p_c adjustment.
3. The presented technique adapts its p_c depending on the search performance in recent iterations, and current generation number. In contrast, most of the previous studies rely the p_c adaptation in the population fitness.

The rest of the paper is structured as follows. In Section **??** the proposed technique is introduced. In Section **??** the problems used in the experimentation are described. Then, in Section **??** the experimentation conducted is shown. In the same section the results obtained by the presented AMCPA are compared with the ones obtained by a basic GA. This work finishes with the conclusions and future work (Section **??**).

2 The proposed Adaptive Multi-Crossover Population Algorithm

As mentioned in the previous section, the proposed AMCPA is a variant of a classical GA. The presented technique reverses the philosophy of conventional GAs, giving higher priority to the individual improvement, provided by the mutation phase. On the other hand, the metaheuristic gives less importance to the crossovers phase and the cooperative improvement. These fundamentals are based on the recently published work (**?**), in which the suitability of some blind crossover operators in GAs for solving path-encoded routing problems is analyzed. In that research study, a theory which stands that the crossover phase is not efficient for the optimization capacity of a GA when it is applied to path-encoded routing problems is checked. For this reason, the proposed metaheuristic provides greater importance to the mutation phase. Despite this, as can be read in (**?**), the crossovers between different individuals can be beneficial to maintain the diversity of the population. Accordingly, in the proposed AMCPA a low p_c is used, which is adapted every generation depending on the search needs. This adaptive mechanism is described in Section **??**.
Besides that, as an additional tool to avoid the premature convergence, a multi-crossover mechanism has been developed, which changes the crossover operator for all the population. These changes are made based on various concepts which are explained in Section **??**. The flowchart of the proposed AMCPA can be seen in Figure **??**. In this figure, green blocks represent the basic steps of the classic GA, also applicable in the presented AMCPA. Furthermore, the blue blocks depict the steps of the adaptive mechanism (Section **??**). Finally, the purple blocks represent the steps of the multi-crossover mechanism (Section **??**). Additionally, the pseudocode of the metaheuristic is depicted in Algorithm **??**.

2.1 Adaptive Mechanism

In the proposed AMCPA every individual in the population goes through the mutation process at each generation. This fact would be equivalent to have a p_m equals to 100%. In addition,

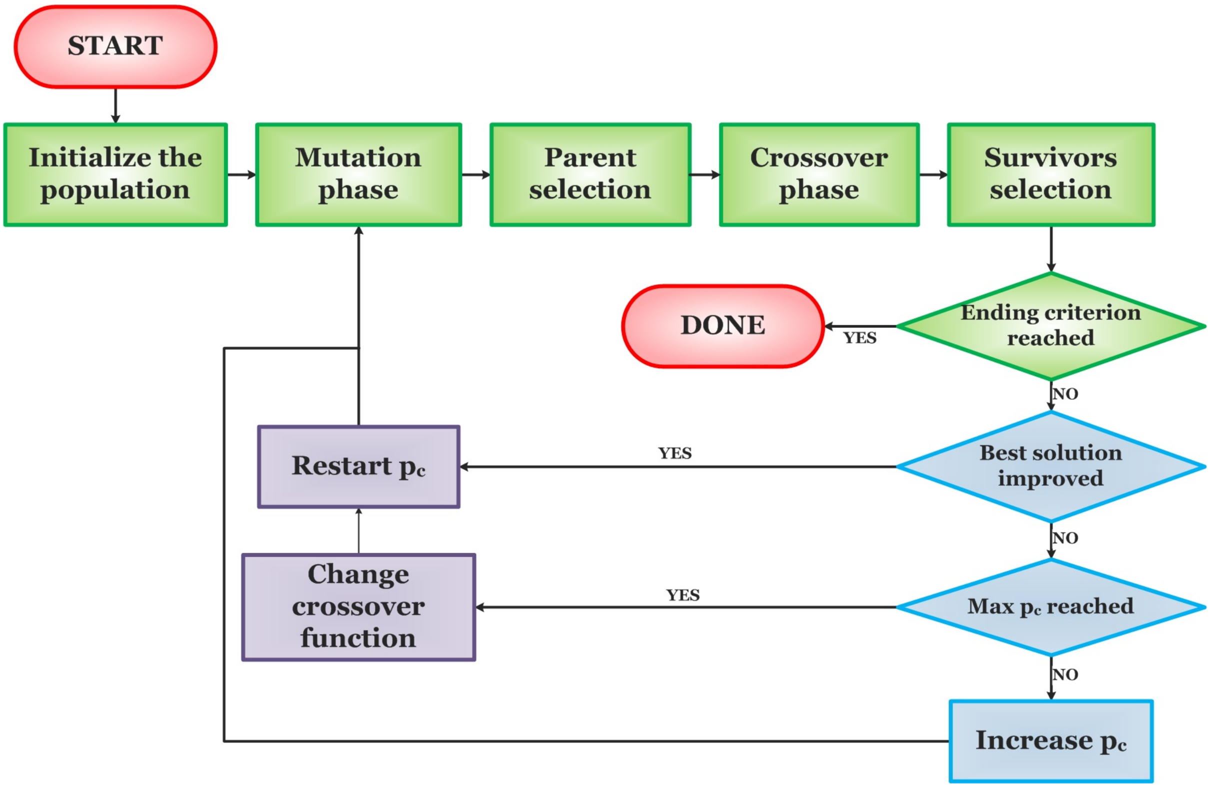

Figure 1: Flowchart of the algorithm

this mutation probability is a fixed value, and it does not vary along the execution. On the other hand, regarding the p_c, it starts with a null (0%) value. That parameter is modified as the algorithmic procedure progresses, increasing or restarting its value. This modification is performed based on the improvement in the best solution found in the last generation. The criteria to modify the p_c are as follows:

- *The best solution found by the technique has been improved in the last generation*: in this case, it could be assumed that it is not necessary to diversify the population. In this case, p_c is restarted to its initial value.
- *The best solution found by the technique has not been improved in the last generation*: in this instance, it may be considered 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, p_c is increased, with the intention of increasing the population diversification using crossover operators.

Whenever the best solution found has not been improved over the previous generation, p_c increases based on the following function:

$$p_c = p_c + \frac{2 \cdot G_{wi} + G}{N_I^3} \tag{2.1}$$

where:

Algorithm 1: Pseudocode of the proposed AMCPA

```
Initialization of the population;
p_c = 0.0;
p_m = 1.0;
repeat
    Mutation phase;
    Parents selection process;
    Crossover phase;
    Survivor selection process;
    if best solution has been improved then
        p_c is restarted;
    else
        if Maxp_c has been reached then
            Change the crossover function;
            p_c is restarted;
        else
            p_c is increased;
        end
    end
until termination criterion reached;
Return the best individual found;
```

G_{wi}: number of generations executed without improvements,
G: total number of generations executed,
N_I: number of individuals in the population,

As seen in Eq. (1), p_c increases proportionally to the number of generations without any improvement in the best solution (G_{wi}) and the total number of generations (G). In Section **??** two examples of its calculation are shown.

2.2 Multi-Crossover mechanism

Regarding the multi-crossover feature, as mentioned in Section **??**, the proposed technique has more than one crossover operator which are alternated during the execution of the algorithm. At the initialization phase, one operator is assigned at random. Then, when necessary, this function is replaced at random by another available, allowing repetition. For this purpose, a maximum value for p_c is defined, $Maxp_c$. If over the generations the p_c value exceeds $Maxp_c$, the crossover function is randomly replaced by another one, and p_c is restarted to its initial value.
It is noteworthy that $Maxp_c$ is an adjustable parameter, which has to be high enough to prevent a premature function change. Additionally, its value cannot be too high, in order to avoid an excessive runtime waste.

It is expected that the multi-crossover mechanism facilitates the population diversification in an efficient way. In this way, it can prevent the search from being trapped in a local optimum. This feature will be tested in Section 4.

3 Descripion of the problems

As has been said in Section **??**, the proposed AMCPA has been applied to six different combinatorial optimization problems. In this section these problems are briefly described. The problems used are the following: Symmetric and Asymmetric Traveling Salesman Problem (TSP and ATSP) (Section **??**), Capacitated Vehicle Routing Problem (CVRP) (Section **??**), Vehicle Routing Problem with Backhauls (VRPB) (Section **??**), N-Queens (NQP) (Section **??**), and the one-dimensional Bin Packing Problem (BPP) (Section **??**).

3.1 Symmetric and Asymmetric Traveling Salesman Problem

The TSP is one of the most famous and widely studied problems throughout history in operations research and computer science. It has a great scientific interest, and it is used in a large number of research studies annually (**?**). This problem can be defined as a complete graph $G = (V, A)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of vertexes which represents the nodes of the system, and $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is the set of arcs which represents the interconnection between nodes. Additionally, each arc has an associated distance cost d_{ij}. In the symmetric version of the TSP the distance between two nodes is the same in both directions, i.e., d_{ij} = d_{ji}. On the other hand, for the ATSP, although there may be pairs of nodes where d_{ij} = d_{ji}, in most cases $d_{ij} \neq d_{ji}$.
The objective of the TSP and ATSP is to find a route that, starting and finishing at the same node, visits every customer once, and that minimizes the total distance traveled. In this way, the objective function for these problems is the total distance traveled in the route.
In this paper, the solutions for the TSP and ATSP are encoded using the well-known path representation (**?**). Thereby, each individual is encoded by a permutation of numbers, which represents the order in which the nodes are visited. For example, for a possible 8-node instance of the TSP, or ATSP, one possible solution would be encoded as $X = (1, 3, 5, 7, 8, 4, 2, 6)$, and its fitness would be $f(X)$ = d_{13}+d_{35}+d_{57}+d_{78}+d_{84}+d_{42}+d_{26}+d_{61}.

3.2 Capacitated Vehicle Routing Problem

The CVRP is also one of the most studied problems in operational research and computers science. Due to its applicability to real life, and its complexity, the CVRP is used in many studies every year (**?**). The CVRP can be defined as a complete graph $G = (V, A)$, where $V = \{v_0, v_1, \ldots, v_n\}$ is the set of vertexes and $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is the set of arcs. The vertex v_0 represents the depot, and the rest are the customers, each of them with a fixed demand q_i. A fleet of vehicles K is available with a limited capacity Q for each of them. The objective of the CVRP is to find a number of routes with a minimum cost such that *1)* each route starts and ends at the depot, *2)* each client is visited exactly by one route and *3)* the total

demand of the customers visited by one route does not exceed the total capacity of the vehicle that performs it (**?**).
In this case, the path representation is also used for the individuals encoding (**?**). In this way, the routes are also represented as a permutation of nodes. In addition, to distinguish the different routes in a solution, they are separated by zeros. For example, in a 8-noded instance, one possible solution of three routes would be encoded as $X = (2, 5, 4, \mathbf{0}, 6, 1, \mathbf{0}, 8, 3, 4)$, and its fitness would be $f(X)$ = d_{02}+d_{25}+d_{54}+d_{40}+ d_{06}+d_{61}+ d_{10}+d_{08}+d_{83}+d_{34}+d_{40}.

3.3 Vehicle Routing Problem with Backhauls

The Vehicle Routing Problem with Backhauls or VRPB is a variant of the basic VRP where customers can demand either a delivery or a pickup of certain goods (**?**). In the VRPB, deliveries are done first, and then the pick-up. This is so because, otherwise, it could be a movement of material within the mobile unit that could be counterproductive. For example, putting materials on the front of the trunk when at the bottom are still some goods that they have not been delivered yet. Thanks to its fidelity to the real world, the VRPB is used in many studies annually (**?**).
This problem can be defined as the CVRP, with the difference that the set of customer V can be separated into two subsets (**?**). The first one, L, called *linehaul customers*, contains those who demand the delivery of goods. On the other hand, the second subset, B, called *backhaul customers*, demand the pickup of a certain amount of material. To express customer demand, a simple way is to use positive values for linehaul customers, and negative values for backhaul ones.
Finally, the path representation is also used for this problem, and the routes are also encoded as nodes permutation. As an example, suppose a set of six linehaul customers $L = \{L1, L2, L3, L4, L5, L6\}$, and six backhaul customers $B = \{B1, B2, B3, B4, B5, B6\}$. One possible solution with three vehicles would be X=($L2$, $L5$, $B1$, $B6$, **0**, $L1$, $L6$, $L4$, $B3$, **0**, $L3$, $B2$, $B5$, $B4$), and it fitness would be $f(X)$ = d_{0L2}+d_{L2L5}+d_{L5B1}+d_{B1B6}+d_{B60} + d_{0L1}+d_{L1L6}+ d_{L6L4}+d_{L4B3}+d_{B30} + d_{0L3}+d_{L3B2}+d_{B2B5}+d_{B5B4}+d_{B40}.

3.4 N-Queens Problem

The NQP is a generalization of the problem of putting eight non attacking queens on a chessboard (**?**), which was introduced by M. Bezzel in 1848 (**?**). This problem consists of placing N queens on a NxN chess board, in order that they cannot attack each other. This problem can be formulated as a combinatorial optimization problem (**?**), despite being a classical combinatorial design problem (constraint satisfaction problem). In the present paper, NQP is formulated as a combinatorial optimization problem, where a solution X is coded as a N-tuple $(q_1, q_2, ..., q_n)$, which is a permutation of the N-tuple $(1, 2, ..., N)$. Each q_i represents the row occupied by the queen positioned in the ith column. Vertical and horizontal collisions are avoided using this representation. Therefore, the objective function of the NQP is defined as the number of diagonal collisions along the board. Notice that ith and jth queens collide diagonally if:

$$|i - q_i| = |j - q_j| \quad \forall i, j : \{1, 2, ..., N\}; i \neq j \tag{3.1}$$

Hence, the objective of this problem is to minimize the number of conflicts, being zero the ideal fitness. This same formulation is frequently used in the literature (**?**; **?**).

3.5 One-dimensional Bin Packing Problem

In distribution and production the fact of packing of items into boxes or bins is a daily task. Depending on the shape and size of the items, as well as the form and capacity of bins, a wide amount of different packing problems can be formulated. The BPP is the simplest problem in this field (**?**; **?**), and it is frequently used in the literature as benchmarking problem (**?**; **?**; **?**). The BPP consists in a set of items $I = \{i_1, i_2, \ldots, i_n\}$, each with an associated size s_i, and an infinite number of bins B of an equal capacity q. The objective of the BPP is to pack all the items into a minimum number of bins. Therefore, the objective function is the number of bins, which has to be minimized.
The solutions of this problem are encoded as a permutation of items. To count the number of bins needed for one solution, the size of the items is accumulated in a variable, $accumSize$. When $accumSize$ exceeds q, the number of bins is increased in 1, and $accumSize$ is restarted. For example, in a simple instance of 10 items, each one with a size of 30, and q=90. One possible solution could be $X = \{i_9, i_6, i_1, i_2, i_4, i_{10}, i_8, i_3, i_7, i_5\}$, and its fitness would be 4.

4 Experimentation

In this section the experimentation performed in this study is detailed. As has been mentioned in Section **??**, the technique proposed in this paper is a variant of the classical GA. For that reason, the results obtained by the presented AMCPA are compared with the ones obtained by a traditional GA. Additionally, in the experimentation performed in this study, not only the qualities of the results are compared, but also the convergence behaviour of both techniques. Furthermore, a statictical study is conducted with these results, using the well-known normal distribution z-test. For both algorithms similar functions and parameters have been used, so that the only difference between them is their working way. This method of comparing metaheuristics is the most reliable way to determine which technique gets better results. In Section **??** the parameters used for each metaheuristic are described. Then, in Section **??** the basic aspects of the experimentation are introduced. After that, the results are shown in Section **??**, and they are analyzed in Section **??**.

4.1 Parameters of the algorithms

For all the experiments and problems, an initial population composed by 50 randomly generated individuals is used by both techniques. The parametrization of the GA has been performed based on the concepts outlined in many previously published works (**?**; **?**; **?**). In accordance with these studies, the crossover is considered the main operator of GAs, while

the mutation is a secondary phase. In line with these concepts, for the GA the p_c has been set in 95%, and the p_m in 5%. In the case of the proposed AMCPA, the p_c starts at 0%. When the best solution found is not improved, the p_c increases following the Equation 1, otherwise, it returns to 0%. Moreover, $Maxp_c$ has been stablished in 40%.
In relation to the parents selection criteria, first, each individual of the population is selected as parent with a probability equal to p_c. If one individual is selected for the crossover, the other participant is selected randomly. Regarding the survivor function, a 50% elitist - 50% random function has been used (which means that the half of the population is composed by the best individuals, and the remaining ones are selected randomly). About the ending criteria, the execution of each algorithm finishes when the population converges. This same criteria has been used many time in the literature (**?**). In the present study, the convergence is assumed when there are $n + \sum_{k=1}^{n} k$ generations without improvements in the best solution, where n is the size of the problem.
Same crossover and mutation functions have been used for the TSP, ATSP, NQP, and BPP. Regarding the crossover functions, for the presented AMCPA Order Crossover (OX) (**?**), Modified Order Crossover (MOX) (**?**), Half Crossover (HX) (**?**), and Order Based Crossover (OBX) (**?**) have been used. These functions have been often used in the literature (**?**; **?**; **?**; **?**; **?**; **?**). On the other hand, OX is used as crossover function for the GA. The mutation function for both techniques is the well-known 2-opt (**?**), which has been widely used since its formulation (**?**; **?**).
Furthermore, the crossover functions used for the proposed AMCPA for the CVRP and VRPB are the Half Route Crossover (HRX) and the Half Random Route Crossover (HRRX) (**?**; **?**). 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 HRX, 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 HRRX working way is similar to HRX. 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 HRX.
Regarding the mutation function used for the CVRP and VRPB, the called Vertex Insertion Routes (**?**) has been used for both metaheuristics. This function selects and extracts one random node from a random route. After that, this node is re-inserted in a random position in another randomly selected route. The creation of new routes is possible with this function.

4.2 Description of the experimentation

In this section the basic aspects of the experimentation are introduced. All the tests have been performed on an Intel Core i7 3930 computer, with 3.20 GHz and a RAM of 16 GB. Java has been used as programming language. For the TSP, 22 instances have been used, and they have been obtained from the TSPLIB Benchmark (**?**). In addition, for the ATSP 19 instances have been chosen, obtained from the same benchmark. For the CVRP 15 instances have been utilized. They have been picked from the VRPWeb (*http://neo.lcc.uma.es/vrp*). The first 11 belong to the Christofides/Eilon benchmark (**?**), and the remaining 4 to the Golden et al.

large-scale benchmark (**?**).

For the VRPB 13 instances have been utilized. The first 6 were obtained from the VRPTW Benchmark of Solomon (*http://w.cba.neu.edu/msolomon/problems.htm*). In this case, the time constraints have been removed, but vehicle capacities and the amount of customer demands are retained. Apart from this, the demands nature has been also modified with the aim of creating pickup and deliveries. The remaining 4 instances have been 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.

Regarding the NQP, 15 different instances have been used. The name of each instance describes the number of queens and the size of the chessboard. In this case, the optimum of each instance is not shown, since it is known that it is 0 for all of them. Finally, in relation to the BPP, another 15 instances have been chosen, which have been picked from the Scholl/Klein benchmark (*http://www.wiwi.uni-jena.de/entscheidung/binpp/index.htm.*). These cases are named $NxCyWz\_a$, where x is 1 (50 items), 2 (100 items), 3 (200 items) or 4 (500 items); y is 1 (capacity of 100), 2 (capacity of 120) and 3 (capacity of 150); z is 1 (items size between 1 and 100) and 2 (items size between 20 and 100); and a is A or B as benchmark indexing parameter.

For each instance 40 runs have been executed, and for each problem, the average fitness value, average runtime, and convergence behaviour are shown. Additionally, the standard deviations of the results and the convergence behaviour are also shown. Furthermore, with the aim of performing a fair and rigorous comparison, the well-known normal distribution z-test has been performed for all experiments. Thanks to this statistical test, it can be demonstrated whether the differences in the results and convergence behaviour of both metaheuristics are significant or not. The z statistic has the following form:

$$z = \frac{\overline{X_{AMCPA}} - \overline{X_{GA}}}{\sqrt{\frac{\sigma^2_{AMCPA}}{n_{AMCPA}} + \frac{\sigma^2_{GA}}{n_{GA}}}}$$

Where $\overline{X_i}$ depicts the average fitness obtained by the technique i, σ_i is the standard deviation of the technique i, and n_i the sample size for technique i. In this way, the value of z can be positive (+), negative (-), or neutral (*). A + indicates that the proposed AMCPA is significantly better. In the opposite case, it obtains substantially worse solutions. If z is *, the difference is not significant. The confidence interval has been stated at 95% ($z_{0.05}$ = 1.96). It is noteworthy that the z-test has been performed for the results quality and convergence.

4.3 Results

In this section the results obtained in the experimentation are shown. As has been mentioned, the tables present the averages and standard deviations of the convergence and results quality. The convergence value depicts the generation in which the technique reaches the final result, and it is displayed in hundreds. Additionally, the average runtimes (in seconds) are also presented. Furthermore, for each instance the results of both z-tests are shown. In Table **??**

Instance		Proposed AMCPA					Genetic Algorithm					z-test	
		Results		Convergence		Time	Results		Convergence		Time		
Name	Optima	Avg.	S. dev.	Avg.	S. dev.	Avg.	Avg.	S. dev.	Avg.	S. dev.	Avg.	resu.	conv.
Oliver30	420	**425.3**	7.6	**6.6**	1.54	0.16	431.4	13.5	**6.6**	3.21	0.25	+	*
Eilon50	425	**439.1**	6.2	**22.0**	6.16	0.35	458.9	16.2	25.8	8.16	1.34	+	+
Eil51	426	**443.4**	10.8	**23.0**	5.71	0.38	460.6	17.3	23.6	8.62	1.37	+	*
Berlin52	7542	**7835.5**	249.5	**10.8**	3.74	0.32	8057.3	194.6	15.4	8.49	1.05	+	+
St70	675	**706.8**	16.0	**49.0**	13.08	1.03	745.1	39.0	83.1	25.09	4.98	+	+
Eilon75	535	**571.4**	10.9	**59.0**	23.47	1.41	614.3	42.7	84.7	27.39	6.42	+	+
Eil76	538	**571.0**	8.7	**50.7**	12.27	1.34	607.3	21.5	**50.7**	14.73	6.82	+	*
KroA100	21282	**22120.1**	520.2	**67.1**	15.77	2.61	22390.4	488.7	72.1	25.84	10.13	+	*
KroB100	22140	**23060.6**	327.8	**71.1**	13.63	2.22	23437.0	598.8	82.1	25.77	10.98	+	+
KroC100	20749	**21670.8**	424.6	**75.7**	16.30	2.35	22394.1	816.0	78.4	32.18	10.65	+	*
KroD100	21294	**22213.2**	382.6	**74.5**	14.72	2.30	23204.8	666.8	77.9	23.78	10.54	+	*
KroE100	22068	**22992.5**	347.0	83.6	22.13	2.75	23289.5	580.5	**79.7**	23.92	10.70	+	*
Eil101	629	**673.4**	9.6	**114.0**	2.71	3.87	713.9	27.1	136.1	41.23	17.33	+	+
Pr107	44303	**45412.3**	699.5	98.8	29.43	3.38	46593.9	1390.0	**95.4**	37.15	15.27	+	*
Pr124	59030	**60493.0**	957.2	104.9	21.03	4.85	62046.3	1538.4	**101.6**	26.46	21.34	+	*
Pr136	96772	**101640.0**	1359.3	**108.6**	23.27	5.63	103963.4	1108.9	115.4	28.93	25.15	+	*
Pr144	58537	**60302.5**	1417.5	**131.1**	10.64	6.40	61884.3	1458.5	136.1	30.01	31.87	+	*
Pr152	73682	**76181.2**	922.0	**166.5**	31.10	8.06	77546.6	1763.5	186.5	59.93	46.86	+	*
Pr264	49135	**53647.3**	1957.5	173.0	27.13	20.54	58117.1	3707.6	**152.8**	31.61	86.05	+	-
Pr299	48191	**55032.7**	4329.8	200.6	52.27	49.75	57331.0	5537.8	**176.7**	49.34	118.33	+	-
Pr439	107217	**117799.2**	3043.6	**736.4**	410.04	97.11	128181.7	27291.3	944.0	507.70	315.05	+	+
Pr1002	259047	**286903.2**	2494.6	**6940.5**	1950.06	315.11	300669.8	11240.1	7215.0	2197.44	821.53	+	*

Table 1: Results of the proposed AMCPA and GA for the TSP

the outcomes for the TSP are depicted. Meanwhile, in Table **??**, the results obtained by both metaheuristics for the CVRP are shown. On the other hand, Table **??** presents the outcomes for the VRPB. Furthermore, results for the ATSP are placed in Table **??**. Moreover, Table **??** and Table **??** display the outcomes for the NQP and BPP, respectively.
Finally, the normal distribution z-test is shown in Table **??**. In this table the statistical tests performed for all the problems and for both parameters (results and convergence behaviour) are depicted.

4.4 Analysis of the results

A clear conclusion can be drawn from the results shown above: the presented AMCPA outperforms the GA in terms of solution quality and runtimes in all the problems used. The AMCPA gets better results in 89.89% of the instances (89 out of 99). In 1 instance (NQP, 8-Queens) the results obtained by both algorithms are the same. Finally, in the remaining 9.09% (9 out of 99), the GA outperforms the AMCPA. Additionally, looking at Table **??**, it can be concluded that these differences are significantly better for the proposed technique in the 85.85% of the cases (85 out of 99). In the 11.11% (11 out of 99) these differences are not substantial, and they are significantly worse only in the remaining 3.03%. On the other hand, the proposed methods needs less runtime in the 96.96% of the cases (96 out of 99), increasing the differences when the size of the instances grows.

		Proposed AMCPA					Genetic Algorithm						
Instance		Results		Convergence		Time	Results		Convergence		Time	z-test	
Name	Optima	Avg.	S. dev.	Avg.	S. dev.	Avg.	Avg.	S. dev.	Avg.	S. dev.	Avg.	resu.	conv.
En22k4	375	395.8	4.6	**20.3**	16.41	0.76	**392.6**	15.3	37.3	36.08	1.22	*	+
En23k3	569	**604.9**	30.0	**55.0**	41.20	0.88	642.5	37.6	94.2	74.57	1.48	+	+
En30k3	503	**541.1**	40.4	87.7	55.88	1.31	578.8	31.2	**84.7**	55.18	2.10	+	*
En33k4	835	**902.6**	27.7	**53.1**	22.97	1.24	917.9	34.0	124.6	66.02	3.68	+	+
En51k5	521	**616.2**	38.1	**136.6**	65.40	3.43	657.0	30.7	174.7	76.29	8.05	+	+
En76k7	682	**812.7**	45.2	**267.2**	125.51	6.42	890.8	40.5	365.6	121.64	27.70	+	+
En76k8	735	**865.7**	34.7	**299**.2	130.03	6.93	951.1	43.8	323.5	158.64	26.19	+	*
En76k10	830	**959.3**	27.0	338.1	180.06	7.98	1031.8	41.8	**306.2**	143.19	27.65	+	*
En76k14	1021	**1143.9**	26.6	**196.3**	98.53	6.51	1205.1	54.6	212.1	74.68	21.50	+	*
En101k8	815	**1003.3**	39.8	486.6	234.97	10.90	1159.7	41.1	**295.0**	98.78	39.49	+	-
En101k14	1071	**1260.4**	58.6	324.2	86.43	11.90	1407.9	53.0	**268.5**	97.28	35.89	+	-
Kelly9	587.09	**844.4**	48.5	**656.5**	191.27	26.74	1112.7	45.4	718.5	21.46	116.44	+	+
Kelly10	746.56	**1117.9**	37.5	**1119.0**	320.36	49.85	1323.6	32.5	1252.9	322.75	176.31	+	*
Kelly11	932.68	**1426.6**	67.2	1817.4	539.80	122.64	1716.1	60.7	**1711.3**	613.12	388.30	+	*
Kelly12	1137.18	**1798.2**	98.3	**1900.7**	422.05	199.07	2108.9	101.4	1954.1	415.42	654.10	+	*

Table 2: Results of the proposed AMCPA and GA for the CVRP

Same conclusions can be extracted by performing an analysis for each problem separately. The presented AMCPA outperforms the GA for the TSP, CVRP, VRPB, NQP, and BPP. Anyway, although the findings are also applicable, it is important to highlight that the differences are narrower for the ATSP. For this problem, the proposed metaheuristic reaches better results in 63.16% of the instances (12 out of 19). Moreover, in the remaining 36.84% it obtains worse results. Additionally, these differentes are significantly better for the AMCPA in 47.36% of the instances, significantly worse in the 15.78%. and not substantial is 36.86%.

The reason why the proposed algorithm needs less runtime is logical, and it can be explained as follows: if crossover and mutation operations are compared, the last one operates with one solution, and it is a simple modification in a chromosome which can be made in a minimum time. Furthermore, the former needs more runtime, since it operates with two different individuals, and its working way is more complex. The proposed AMCPA makes fewer crossovers than the GA, and this fact is perfectly reflected in runtimes, providing an advantage to the AMCPA.

On the other hand, the reason why the proposed AMCPA gets better results can also be explained, and it is based on the recently published study (**?**). According to that work, crossovers are useful resources to make jumps in the space of solutions when they are applied to combinatorial optimization problems. Thereby, the use of crossovers helps a wide exploration of the solution space, but it does not help to perform an exhaustive search of promising regions. To get a deeper search, a function that takes care of optimizing the solutions independently becomes necessary, in order to conduct small jump in the space of solutions. The mutation function can handle this goal.

Regarding the convergence behaviour, looking at Tables 1-6, it can be said that the proposed AMCPA presents a better convergence behaviour. This fact means that the AMCPA need less generations to reach its final solution. Performing a general analysis, the presented algorithm

	Proposed AMCPA					Genetic Algorithm						
Instance	Results		Convergence		Time	Results		Convergence		Time	z-test	
Name	Avg.	S. dev.	Avg.	S. dev.	Avg.	Avg.	S. dev.	Avg.	S. dev.	Avg.	resu.	conv.
C101	**718.5**	42.7	**190.7**	85.65	8.25	727.8	51.4	198.2	83.34	20.08	*	*
C201	**620.2**	42.2	**200.2**	98.90	3.91	834.1	30.3	237.7	67.04	4.34	+	+
R101	**935.2**	47.5	**177.2**	62.84	6.28	1033.4	86.3	289.7	155.55	20.50	+	+
R201	**1072.7**	41.6	293.7	115.00	11.20	1307.2	95.0	**245.1**	95.11	29.98	+	-
RC101	**584.3**	45.4	**88.9**	38.19	2.59	685.3	127.9	91.8	41.38	2.69	+	*
RC201	**1191.7**	62.6	**338.3**	106.97	14.44	1438.5	87.4	362.5	100.66	26.98	+	*
En22k4	**386.5**	16.1	**26.1**	22.59	0.98	403.0	18.4	34.7	21.21	0.96	+	*
En23k3	**712.6**	20.7	**22.1**	16.47	0.90	731.5	37.2	27.9	23.43	0.85	+	*
En30k4	**542.0**	37.0	**58.1**	37.67	1.09	594.0	63.6	60.8	34.52	1.21	+	*
En33k4	**818.4**	32.2	**57.4**	37.31	1.59	846.0	35.9	83.3	42.10	2.05	+	+
En51k5	**669.2**	36.4	**90.6**	47.12	2.72	737.1	47.3	99.1	53.22	4.14	+	*
En76k8	**906.2**	47.4	**147.2**	69.56	5.99	996.6	92.9	184.3	72.73	18.52	+	+
En101k14	**1210.8**	30.0	**180.8**	64.49	11.17	1247.2	68.2	469.2	317.37	60.68	+	+

Table 3: Results of the proposed AMCPA and GA for the VRPB

has a better behavior in 70.70% (70 out of 99) of the cases. On the other hand, in the 27.27% of the instances the GA shows a better performance. Finally, in the remaining 2 instances both metaheuristics present the same behaviour.

Anyway, these differences in the convergence behaviour are significantly better for the AMCPA only in the 33.33% of the instances (33 out of 99), being insignificant in 57.57% of the cases (57 out of 99), and substantially worse in the remaining 9.09%. This fact means that, despite of showing a better convergence in 70 instance, this improvement is not remarkable in almost the half of cases (33 out of 70). This is why it can be said that the AMCPA presents a better performance, but is not as distinctive as the improvement in the results discussed above. Even so, in overall, the proposed method outperforms the GA also in this aspect. This fact provides a great advantage to AMCPA, since, thanks to its better exploration capacity, it is able to find the final solution performing less generations and consuming less computational resources.

By way of conclusion, the proposed AMCPA is a metaheuristic perfectly able to perform an intense and thorough search in promising regions of the solution space using the mutation function. Meanwhile, it performs crossover in case the search is in a local optimum, with the aim of escaping local optimums. Using crossovers, the current population is expanded through the entire solution space. In this way, it is easier to find regions that allow the search to reach better results. This diversification is enhanced thanks to the multi-crossover, allowing a wider exploration.

Conversely, with the GA the search performed comprises a large area of the solution space, but it has a smaller capacity to deepen in those areas which are most promising. This fact leads to the GA to obtain worse results than the AMCPA.

5 Conclusions

In this paper a new Adaptive Multi-Crossover Population Algorithm for solving combinatorial optimization problems has been presented, which is a variant of the classical genetic algorithm.

		Proposed AMCPA					Genetic Algorithm						
Instance		Results		Convergence		Time	Results		Convergence		Time	z-test	
Name	Optima	Avg.	S. dev.	Avg.	S. dev.	Avg.	Avg.	S. dev.	Avg.	S. dev.	Avg.	resu.	conv.
br17	39	**39.1**	0.2	**1.0**	0.34	0.04	39.6	1.0	1.2	0.57	0.05	+	*
ftv33	1286	**1385.9**	65.8	8.2	4.74	0.22	1386.2	45.8	**6.4**	2.60	0.25	*	-
ftv35	1473	1569.1	47.7	9.1	6.73	0.28	**1564.3**	49.1	**7.5**	2.67	0.31	*	*
ftv38	1530	**1611.5**	52.7	**11.8**	7.49	0.38	1635.9	62.5	13.8	10.98	0.44	*	*
p43	5620	**5628.3**	6.1	13.4	5.42	0.38	5635.2	10.6	**12.6**	9.42	0.62	+	*
ftv44	1613	1782.2	83.6	17.5	9.87	0.59	**1748.1**	50.2	**14.9**	6.46	0.73	-	*
ftv47	1776	1904.0	96.6	**20.4**	8.86	0.71	**1862.4**	56.2	28.4	12.67	0.99	-	+
ry48p	14422	**14824.9**	177.6	**20.7**	8.32	0.73	15095.3	490.0	21.5	13.81	1.03	+	*
ft53	6905	7777.6	283.8	33.2	17.11	1.44	**7732.3**	428.2	**31.1**	16.01	1.78	*	*
ftv55	1608	**1791.4**	69.9	**29.1**	24.11	1.54	1837.2	89.2	31.5	20.42	1.88	+	*
ftv64	1839	**2081.4**	133.5	**35.7**	16.32	2.22	2112.2	48.4	38.8	19.53	2.93	*	*
ftv70	1950	2204.6	106.3	58.1	25.12	3.98	**2144.4**	95.6	**56.9**	22.21	4.59	-	*
ft70	38673	**40375.3**	436.2	**65.1**	27.24	4.80	40778.9	1041.6	77.9	29.61	5.32	+	+
kro124p	36230	**39009.8**	824.9	**80.1**	35.39	8.89	40063.7	1270.8	84.0	37.32	11.23	+	*
ftv170	2755	4022.9	354.1	150.1	40.31	24.88	**3947.5**	320.3	**142.0**	37.80	39.97	*	*
rbg323	1326	**1901.5**	113.9	**99.3**	38.67	68.01	2123.1	144.5	104.3	36.59	77.68	+	*
rbg358	1163	**1907.9**	152.6	162.1	42.17	97.86	2034.0	165.6	**157.7**	44.69	112.33	+	*
rbg403	2465	**2896.9**	58.1	**175.8**	34.44	210.11	2978.5	79.8	195.2	71.40	259.61	+	*
rbg443	2720	3415.3	137.4	213.0	38.98	289.48	**3380.6**	105.6	**201.6**	45.08	346.27	*	*

Table 4: Results of the proposed AMCPA and GA for the ATSP

The proposed metaheuristic 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, the presented technique has been introduced, explaining how it works. Then, the six problems used and the experimentation have been described. After that, the results obtained by the technique have been shown. These outcomes have been compared with the ones obtained by a classical GA, to conclude that the proposed method gets better results. Finally, why the presented AMCPA is better than the GA has been reasoned.

As future work, it is intended to compare the performance of the introduced technique with other approaches of similar philosophy that can be found in the literature. Furthermore, it is planned to apply the AMCPA to real life routing problems. At this time, it will be applied 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.

Acknowledgment

This work is an extension of the paper ”An Adaptive Multi-Crossover Population Algorithm for Solving Routing Problems”, presented at the VI International Workshop on Nature Inspired Cooperative Strategies for Optimization (NICSO 2013) (**?**).

Instance	Proposed AMCPA					Genetic Algorithm					z-test	
	Results		Convergence		Time	Results		Convergence		Time		
Name	Avg.	S. dev.	Avg.	S. dev.	Avg.	Avg.	S. dev.	Avg.	S. dev.	Avg.	resu.	conv.
8-Queens	**0.0**	0.0	**0.1**	0.03	0.01	**0.0**	0.0	0.2	0.02	0.01	*	+
20-Queens	**0.9**	0.5	0.4	0.43	0.04	1.8	1.1	**0.2**	0.15	0.02	+	-
50-Queens	**5.7**	1.5	**0.7**	0.22	0.13	8.5	1.9	1.0	0.64	0.17	+	+
75-Queens	**11.6**	2.6	**1.7**	2.08	0.21	15.4	3.1	2.2	1.05	0.61	+	*
100-Queens	**15.4**	3.1	**2.1**	0.75	0.58	23.2	3.2	2.7	1.32	1.31	+	+
125-Queens	**19.3**	3.3	**3.5**	1.15	1.39	30.2	3.6	4.2	2.81	2.99	+	*
150-Queens	**22.4**	3.0	**4.9**	1.55	2.71	37.2	4.6	6.3	3.60	5.94	+	+
200-Queens	**32.4**	5.8	**8.2**	2.91	7.69	49.9	5.1	11.8	5.16	18.51	+	+
225-Queens	**39.8**	5.7	**9.2**	3.00	10.85	57.8	6.0	13.0	6.03	25.92	+	+
250-Queens	**43.2**	4.6	**11.3**	3.16	16.42	63.9	6.0	15.4	7.13	36.04	+	+
275-Queens	**47.3**	6.5	**13.6**	2.94	23.48	69.0	9.6	19.9	8.08	56.97	+	+
300-Queens	**51.2**	6.0	**15.3**	3.77	31.83	76.9	6.5	22.4	7.10	76.22	+	+
325-Queens	**56.2**	5.6	**17.4**	3.97	42.37	86.0	9.4	23.9	10.57	99.25	+	+
350-Queens	**60.9**	7.8	**19.7**	5.73	55.32	90.8	11.3	31.6	12.86	143.30	+	+
400-Queens	**72.7**	6.6	**25.5**	5.94	93.33	101.8	10.0	50.7	11.28	319.50	+	+

Table 5: Results of the proposed AMCPA and GA for the NQP

References

Albayrak, M. and Allahverdi, N. 2011. Development a new mutation operator to solve the traveling salesman problem by aid of genetic algorithms, *Expert Systems with Applications* **38**(3): 1313–1320.

Ali, M. Z., Alkhatib, K. and Tashtoush, Y. 2013. Cultural algorithms: Emerging social structures for the solution of complex optimization problems, *International Journal of Artificial Intelligence* **11**(A13): 20–42.

Atashpaz-Gargari, E. and Lucas, C. 2007. Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition, *Proceedings of the Congress on Evolutionary Computation*, IEEE, pp. 4661–4667.

Bell, J. and Stevens, B. 2009. A survey of known results and research areas for n-queens, *Discrete Mathematics* **309**(1): 1–31.

Bezzel, M. 1848. Proposal of 8-queens problem, *Berliner Schachzeitung* **3**: 363.

Bianchessi, N. and Righini, G. 2007. Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery, *Computers & Operations Research* **34**(2): 578–594.

Blum, C., Puchinger, J., Raidl, G. R. and Roli, A. 2011. Hybrid metaheuristics in combinatorial optimization: A survey, *Applied Soft Computing* **11**(6): 4135–4151.

Cantú-Paz, E. 1998. A survey of parallel genetic algorithms, *Calculateurs paralleles, reseaux et systems repartis* **10**(2): 141–171.

Chang, W.-D. 2007. A multi-crossover genetic approach to multivariable pid controllers tuning, *Expert Systems with Applications* **33**(3): 620–626.

Instance		Proposed AMCPA					Genetic Algorithm					z-test	
		Results		Convergence		Time	Results		Convergence		Time		
Name	Optima	Avg.	S. dev.	Avg.	S. dev.	Avg.	Avg.	S. dev.	Avg.	S. dev.	Avg.	resu.	conv.
N1C1W1_A	25	**27.0**	0.3	**0.15**	0.09	0.01	27.4	0.5	0.17	0.07	0.06	+	*
N1C1W1_B	31	**31.8**	0.4	**0.21**	0.22	0.01	32.2	0.7	0.27	0.30	0.04	+	*
N1C2W1_A	21	**21.9**	0.6	0.26	0.25	0.01	22.2	0.5	**0.20**	0.15	0.04	+	*
N1C2W1_B	26	**27.1**	0.3	**0.16**	0.11	0.01	27.6	0.5	0.23	0.24	0.04	+	*
N2C1W1_A	48	**53.1**	0.8	**1.23**	0.74	0.05	53.6	0.7	1.30	1.68	0.27	+	*
N2C1W1_B	49	**53.4**	0.8	0.81	0.38	0.05	53.8	0.4	**0.59**	0.30	0.22	+	-
N2C2W1_A	42	**45.7**	1.0	**0.69**	0.39	0.02	46.3	0.7	0.78	0.47	0.24	+	*
N2C2W1_B	50	**53.7**	0.8	0.81	0.37	0.03	54.1	0.5	**0.69**	0.43	0.22	+	*
N3C2W2_A	107	**120.6**	1.2	**2.55**	1.42	0.10	121.5	1.8	3.02	1.36	1.74	+	*
N3C2W2_B	105	116.2	1.4	**3.04**	1.61	0.11	**116.0**	1.4	3.18	1.79	1.75	*	*
N3C3W1_A	66	**72.8**	0.9	1.47	0.88	0.08	74.1	1.0	**1.33**	0.98	1.26	+	*
N3C3W1_B	71	**79.0**	0.6	**1.30**	0.53	0.08	80.1	2.4	1.49	1.22	1.27	+	*
N4C2W1_A	210	**241.8**	1.7	**10.20**	3.90	1.54	244.7	2.5	13.63	5.92	24.57	+	+
N4C2W1_B	213	**245.2**	1.7	15.62	6.38	1.57	247.8	2.5	**12.47**	7.18	23.61	+	-
N4C2W1_C	213	**245.1**	1.9	13.82	7.18	1.62	247.9	2.3	**12.46**	5.96	23.27	+	*

Table 6: Results of the proposed AMCPA and GA for the BPP

z-test							
Results	TSP	CVRP	VRPB	ATSP	NQP	BPP	Total
+	22	14	12	9	14	14	85
$*$	0	1	1	7	1	1	11
-	0	0	0	3	0	0	3
Convergence	TSP	CVRP	VRPB	ATSP	NQP	BPP	Total
+	7	6	5	2	12	1	33
$*$	13	7	7	16	2	12	57
-	2	2	1	1	1	2	9

Table 7: Summary of the z-test. '+' indicates that AMCPA is better. '-' depicts that it is worse. '*' indicates that the differences are not significant (at 95% confidence level)

Cho, D. W., Lee, Y. H., Lee, T. Y. and Gen, M. 2013. An adaptive genetic algorithm for the time dependent inventory routing problem, *Journal of Intelligent Manufacturing* pp. 1–18.

Christofides, N. and Eilon, S. 1969. An algorithm for the vehicle-dispatching problem, *OR* pp. 309–318.

Coffman, E. G. and Bruno, J. L. 1976. *Computer and job-shop scheduling theory*, John Wiley & Sons.

Cordeau, J. and Maischberger, M. 2012. A parallel iterated tabu search heuristic for vehicle routing problems, *Computers & Operations Research* **39**(9): 2033–2050.

Davis, L. 1985. Applying adaptive algorithms to epistatic domains, *Proceedings of the international joint conference on artificial intelligence*, Vol. 1, pp. 161–163.

Davis, L. 1989. Adapting operator probabilities in genetic algorithms, *Proceeding of the Third International Conference on Genetic Algorithms*, pp. 61–69.

De Giovanni, L., Massi, G. and Pezzella, F. 2013. An adaptive genetic algorithm for large-size open stack problems, *International Journal of Production Research* **51**(3): 682–697.

De Jong, K. 1975. *Analysis of the behavior of a class of genetic adaptive systems*, PhD thesis, University of Michigan, Michigan, USA.

Dorigo, M. and Gambardella, L. M. 1997. Ant colony system: A cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computation* **1**(1): 53–66.

Eberhart, R. C. and Shi, Y. 2001. Particle swarm optimization: developments, applications and resources, *Proceedings of the Congress on Evolutionary Computation*, Vol. 1, IEEE, pp. 81–86.

Fernandez-Prieto, J., Gadeo-Martos, M. and Velasco, J. R. 2011. Optimisation of control parameters for genetic algorithms to test computer networks under realistic traffic loads, *Applied Soft Computing* **11**(4): 3744–3752.

Fleszar, K. and Charalambous, C. 2011. Average-weight-controlled bin-oriented heuristics for the one-dimensional bin-packing problem, *European Journal of Operational Research* **210**(2): 176–184.

Fogel, D. B. 1994. An introduction to simulated evolutionary optimization, *IEEE Transactions on Neural Networks* **5**(1): 3–14.

Gao, J., Gen, M., Sun, L. and Zhao, X. 2007. A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems, *Computers & Industrial Engineering* **53**(1): 149–162.

Glover, F. 1989. Tabu search, part i, *ORSA Journal on computing* **1**(3): 190–206.

Goldberg, D. 1989. *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley Professional.

Golden, B., Baker, E., Alfaro, J. and Schaffer, J. 1985. The vehicle routing problem with backhauling: two approaches, *Proceedings of the Twenty-first Annual Meeting of SE TIMS*, South Carolina USA, pp. 90–92.

Golden, B. L., Raghavan, S. and Wasil, E. A. 2008. *The Vehicle Routing Problem: Latest Advances and New Challenges*, Vol. 43, Springer.

Golden, B. L., Wasil, E. A., Kelly, J. P. and Chao, I.-M. 1998. The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results, *Fleet management and logistics*, Kluwer, Boston, pp. 33–56.

Grefenstette, J. J. 1986. Optimization of control parameters for genetic algorithms, *IEEE Transactions on Systems, Man and Cybernetics* **16**(1): 122–128.

Holland, J. H. 1975. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, MIT press.

Hu, X., Eberhart, R. C. and Shi, Y. 2003. Swarm intelligence for permutation optimization: a case study of n-queens problem, *Proceedings of the IEEE Swarm Intelligence Symposium*, pp. 243–246.

Joelianto, E. and Wiranto, I. 2011. An application of ant colony optimization, kalman filter and artificial neural network for multiple target tracking problems, *International Journal of Artificial Intelligence* **7**(A11): 384–400.

Karaboga, D., Gorkemli, B., Ozturk, C. and Karaboga, N. 2012. A comprehensive survey: artificial bee colony (abc) algorithm and applications, *Artificial Intelligence Review* pp. 1–37.

Karmarkar, N. and Karp, R. M. 1982. An efficient approximation scheme for the one-dimensional bin-packing problem, *23rd Annual Symposium on Foundations of Computer Science*, IEEE, pp. 312–320.

Kennedy, J. and Eberhart, R. 1995. Particle swarm optimization, *Proceedings of IEEE international conference on neural networks*, Vol. 4, Perth, Australia, pp. 1942–1948.

Kirkpatrick, S., Gellat, C. and Vecchi, M. 1983. Optimization by simmulated annealing, *Science* **220**(4598): 671–680.

Knysh, D. and Kureichik, V. 2010. Parallel genetic algorithms: a survey and problem state of the art, *Journal of Computer and Systems Sciences International* **49**(4): 579–589.

Larranaga, P., Kuijpers, C. M. H., Murga, R. H., Inza, I. and Dizdarevic, S. 1999. Genetic algorithms for the travelling salesman problem: A review of representations and operators, *Artificial Intelligence Review* **13**(2): 129–170.

Lawler, E., Lenstra, J., Kan, A. and Shmoys, D. 1985. *The traveling salesman problem: a guided tour of combinatorial optimization*, Vol. 3, Wiley New York.

Lenstra, J. and Kan, A. 1981. Complexity of vehicle routing and scheduling problems, *Networks* **11**(2): 221–227.

Lin, S. 1965. Computer solutions of the traveling salesman problem, *Bell System Technical Journal* **44**(10): 2245–2269.

Martello, S. and Toth, P. 1990. *Knapsack problems*, Wiley New York.

Martinjak, I. and Golub, M. 2007. Comparison of heuristic algorithms for the n-queen problem, *29th IEEE International Conference on Information Technology Interfaces*, pp. 759–764.

Masehian, E., Akbaripour, H. and Mohabbati-Kalejahi, N. 2013. Landscape analysis and efficient metaheuristics for solving the n-queens problem, *Computational Optimization and Applications* **56**(3): 735–764.

Moon, I., Lee, J.-H. and Seong, J. 2012. Vehicle routing problem with time windows considering overtime and outsourcing vehicles, *Expert Systems with Applications* **39**(18): 13202–13213.

Moraglio, A. and Poli, R. 2011. Geometric crossover for the permutation representation, *Intelligenza Artificiale* **5**(1): 49–63.

Mukherjee, S., Ganguly, S. and Das, S. 2012. A strategy adaptive genetic algorithm for solving the travelling salesman problem, *Swarm, Evolutionary, and Memetic Computing*, Springer, pp. 778–784.

Nasiri, B. and Meybodi, M. 2012. Speciation based firefly algorithm for optimization in dynamic environments, *International Journal of Artificial Intelligence* **8**(S12): 118–132.

Osaba, E., Carballedo, R., Diaz, F. and Perallos, A. 2013. Analysis of the suitability of using blind crossover operators in genetic algorithms for solving routing problems, *Proceedings of the 8th International Symposium on Applied Computational Intelligence and Informatics*, IEEE, pp. 17–23.

Osaba, E., Diaz, F. and Onieva, E. 2013. A novel meta-heuristic based on soccer concepts to solve routing problems, *Proceeding of the fifteenth annual conference companion on Genetic and evolutionary computation*, ACM, pp. 1743–1744.

Osaba, E., Diaz, F. and Onieva, E. 2014. Golden ball: a novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts, *Applied Intelligence* **41**(1): 145–166.

Osaba, E., Onieva, E., Carballedo, R., Diaz, F. and Perallos, A. 2014. An adaptive multi-crossover population algorithm for solving routing problems, *Proceedings of the 6th International Workshop on Nature Inspired Cooperative Strategies for Optimization*, Springer, pp. 113–124.

Ponz-Tienda, J. L., Yepes, V., Pellicer, E. and Moreno-Flores, J. 2013. The resource leveling problem with multiple resources using an adaptive genetic algorithm, *Automation in Construction* **29**: 161–172.

Precup, R.-E., David, R.-C., Petriu, E. M., Preitl, S. and Paul, A. S. 2011. Gravitational search algorithm-based tuning of fuzzy control systems with a reduced parametric sensitivity, *Soft Computing in Industrial Applications*, Springer, pp. 141–150.

Prins, C. 2004. A simple and effective evolutionary algorithm for the vehicle routing problem, *Computers & Operations Research* **31**(12): 1985–2002.

Purcaru, C., Precup, R.-E., Iercan, D., Fedorovici, L.-O., David, R.-C. and Dragan, F. 2013. Optimal robot path planning using gravitational search algorithm, *International Journal of Artificial Intelligence* **10**(S13): 1–20.

Rashedi, E., Nezamabadi-Pour, H. and Saryazdi, S. 2009. Gsa: a gravitational search algorithm, *Information sciences* **179**(13): 2232–2248.

Ray, S., Bandyopadhyay, S. and Pal, S. 2004. New operators of genetic algorithms for traveling salesman problem, *Proceedings of the 17th International Conference on Pattern Recognition*, Vol. 2, IEEE, pp. 497–500.

Rego, C., Gamboa, D., Glover, F. and Osterman, C. 2011. Traveling salesman problem heuristics: leading methods, implementations and latest advances, *European Journal of Operational Research* **211**(3): 427–441.

Reinelt, G. 1991. Tsplib: A traveling salesman problem library, *ORSA journal on computing* **3**(4): 376–384.

Reynolds, R. G. 1994. An introduction to cultural algorithms, *Proceedings of the third annual conference on evolutionary programming*, Singapore, pp. 131–139.

Rocha, M., Sousa, P., Cortez, P. and Rio, M. 2011. Quality of service constrained routing optimization using evolutionary computation, *Applied Soft Computing* **11**(1): 356–364.

Salhi, S., Wassan, N. and Hajarat, M. 2013. The fleet size and mix vehicle routing problem with backhauls: Formulation and set partitioning-based heuristics, *Transportation Research Part E: Logistics and Transportation Review* **56**: 22–35.

Schaffer, J. D. and Morishima, A. 1987. An adaptive crossover distribution mechanism for genetic algorithms, *Proceedings of the Second International Conference on Genetic algorithms and their application*, L. Erlbaum Associates Inc., pp. 36–40.

Sharma, S. and Gupta, K. 2011. Solving the traveling salesmen problem through genetic algorithm with new variation order crossover, *International Conference on Emerging Trends in Networks and Computer Communications*, IEEE, pp. 274–276.

Sim, K. and Hart, E. 2013. Generating single and multiple cooperative heuristics for the one dimensional bin packing problem using a single node genetic programming island model, *Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*, ACM, pp. 1549–1556.

Sim, K., Hart, E. and Paechter, B. 2012. A hyper-heuristic classifier for one dimensional bin packing problems: Improving classification accuracy by attribute evolution, *Proceeding of the XII conference on Parallel Problem Solving from Nature*, Springer, pp. 348–357.

Spears, W. M. 1995. Adapting crossover in evolutionary algorithms, *Proceedings of the Conference on Evolutionary Programming*, pp. 367–384.

Srinivas, M. and Patnaik, L. M. 1994a. Adaptive probabilities of crossover and mutation in genetic algorithms, *IEEE Transactions on Systems, Man and Cybernetics* **24**(4): 656–667.

Srinivas, M. and Patnaik, L. M. 1994b. Genetic algorithms: A survey, *Computer* **27**(6): 17–26.

Syswerda, G. 1991. Schedule optimization using genetic algorithms, *Handbook of genetic algorithms* pp. 332–349.

Tarantilis, C. and Kiranoudis, C. 2007. 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): 806–822.

Tomassini, M. 1995. A survey of genetic algorithms, *Annual Reviews of Computational Physics* **3**(2): 87–118.

Toth, P. and Vigo, D. 2002a. *The vehicle routing problem*, Vol. 9, Society for Industrial & Applied Mathematics, SIAM, Philadelphia.

Toth, P. and Vigo, D. 2002b. Vrp with backhauls, *The Vehicle Routing Problem, SIAM Monographs on Discrete Mathematics and Applications* **9**: 195–221.

Vafaee, F. and Nelson, P. C. 2009. A genetic algorithm that incorporates an adaptive mutation based on an evolutionary model, *Proceedings of the International Conference on Machine Learning and Applications*, IEEE, pp. 101–107.

Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N. and Rei, W. 2012. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems, *Operations Research* **60**(3): 611–624.

Wang, C., Zhang, J., Yang, J., Hu, C. and Liu, J. 2005. A modified particle swarm optimization algorithm and its application for solving traveling salesman problem, *Proceedings of the International Conference on Neural Networks and Brain*, Vol. 2, IEEE, pp. 689–694.

Wang, L. and Tang, D.-b. 2011. An improved adaptive genetic algorithm based on hormone modulation mechanism for job-shop scheduling problem, *Expert Systems with Applications* **38**(6): 7243–7250.

Xing, B. and Gao, W.-J. 2014. Imperialist competitive algorithm, *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms*, Springer, pp. 203–209.

Xu, P., Zheng, J., Chen, H. and Liu, P. 2010. Optimal design of high pressure hydrogen storage vessel using an adaptive genetic algorithm, *International Journal of Hydrogen Energy* **35**(7): 2840–2846.

Yang, K., Zheng, J., Yang, M., Zhou, R. and Liu, G. 2013. Adaptive genetic algorithm for daily optimal operation of cascade reservoirs and its improvement strategy, *Water resources management* **27**(12): 4209–4235.

Yang, X.-S. 2010. Firefly algorithm, stochastic test functions and design optimisation, *International Journal of Bio-Inspired Computation* **2**(2): 78–84.

Yazdani, D., Nasiri, B., Azizi, R., Sepas-Moghaddam, A. and Meybodi, M. R. 2013. Optimization in dynamic environments utilizing a novel method based on particle swarm optimization, *International Journal of Artificial Intelligence* **11**(A13): 170–192.

Ye, Z., Li, Z. and Xie, M. 2010. Some improvements on adaptive genetic algorithms for reliability-related applications, *Reliability Engineering & System Safety* **95**(2): 120–126.

Zhang, J., Chung, H. S. and Lo, W.-L. 2007. Clustering-based adaptive crossover and mutation probabilities for genetic algorithms, *IEEE Transactions on Evolutionary Computation* **11**(3): 326–335.

Zhang, J., Chung, H. S. and Zhong, J. 2005. Adaptive crossover and mutation in genetic algorithms based on clustering technique, *Proceedings of the Conference on Genetic and Evolutionary Computation*, ACM, pp. 1577–1578.

Zhang, X. and Yuen, S. Y. 2013. Improving artificial bee colony with one-position inheritance mechanism, *Memetic Computing* **5**(3): 187–211.

Zhang, Z., Zhang, S., Wang, Y., Jiang, Y. and Wang, H. 2013. Use of parallel deterministic dynamic programming and hierarchical adaptive genetic algorithm for reservoir operation optimization, *Computers & Industrial Engineering* **65**(2): 310–321.