Citation detail: Osaba, Eneko and Yang, Xin-She and Diaz, Fernando and Lopez-Garcia, Pedro and Carballedo, Roberto (2015) An improved discrete bat algorithm for symmetric and asymmetric Traveling Salesman Problems. *Engineering Applications of Artificial Intelligence*, 48(1), 59-71 (2016).

An Improved Discrete Bat Algorithm for Symmetric and Asymmetric Traveling Salesman Problems

Eneko Osaba[a,b,*], Xin-She Yang[b], Fernando Diaz[a], Pedro Lopez-Garcia[a], Roberto Carballedo[a]

[a]*Deusto Institute of Technology (DeustoTech), University of Deusto, Av. Universidades 24, Bilbao 48007, Spain*
[b]*School of Science and Technology, Middlesex University, Hendon Campus, London, NW4 4BT, United Kingdom*

Abstract

Bat algorithm is a population metaheuristic proposed in 2010 which is based on the echolocation or bio-sonar characteristics of microbats. Since its first implementation, the bat algorithm has been used in a wide range of fields. In this paper, we present a discrete version of the bat algorithm to solve the well-known symmetric and asymmetric traveling salesman problems. In addition, we propose an improvement in the basic structure of the classic bat algorithm. To prove that our proposal is a promising approximation method, we have compared its performance in 37 instances with the results obtained by five different techniques: evolutionary simulated annealing, genetic algorithm, an island based distributed genetic algorithm, a discrete firefly algorithm and an imperialist competitive algorithm. In order to obtain fair and rigorous comparisons, we have conducted three different statistical tests along the paper: the Student's t-test, the Holm's test, and the Friedman test. We have also compared the convergence behaviour shown by our proposal with the ones shown by the evolutionary simulated annealing, and the discrete firefly algorithm. The experimentation carried out in this study has shown that the presented improved bat algorithm outperforms significantly all the other alternatives in most of the cases.

Keywords: Bat Algorithm, Traveling Salesman Problem, Genetic Algorithms, Combinatorial Optimization, Routing Problems

*Corresponding author
Email addresses: `e.osaba@deusto.es`, `Eneko1@mdx.ac.uk` (Eneko Osaba), `x.yang@mdx.ac.uk` (Xin-She Yang), `fernando.diaz@deusto.es` (Fernando Diaz), `p.lopez@deusto.es` (Pedro Lopez-Garcia), `roberto.carballedo@deusto.es` (Roberto Carballedo)

1. Introduction

Combinatorial optimization is one of the most studied fields in artificial intelligence, optimization, logistics, and other applications. Multiple research works are published annually in this area, both in journals (Kasperski & Zieliński (2015)), and conferences (Bezerra et al. (2014)), and also in books (Levin (2015)). Different sort of problems exist within this kind of optimization, including the routing problems as one of the most appealed. It is noteworthy that the most used and well-known routing problems are the Traveling Salesman Problem (TSP) (Lawler et al. (1985)), and the Vehicle Routing Problem (VRP) (Christofides (1976)), which are the focus of a huge amount of studies in the literature (Groba et al. (2015); Bortfeldt et al. (2015)).

The main reasons for the popularity and importance of the routing problems are two folds: the social interest they generate, and their inherent scientific interest. On the one hand, routing problems are normally designed to deal with real world situations related to the transport or logistics. This is the reason why their efficient resolution entails a profit, either social or business one. On the other hand, most of the problems arising in this field have a great computational complexity. Being NP-Hard, the resolution of these problems is a major challenge for the scientific community.

In line with this, diverse appropriate approaches can be found in the literature to address this kind of problems efficiently. Arguably the most successful techniques are the exact methods (Laporte (1992a,b)), heuristics and metaheuristics. In this paper, we focus our attention in the last ones. Some classical examples of metaheuristics can be the simulated annealing (SA) (Kirkpatrick et al. (1983)), and the tabu search (Glover (1989)), as local search-based methods, and genetic algorithm (GA) (Goldberg (1989); De Jong (1975)), particle swarm optimization (Kennedy et al. (1995); Tang et al. (2015)), and ant colony optimization (Dorigo & Blum (2005)) as population-based ones. Despite having been proposed many years ago, these techniques remain successful in scientific community nowadays, being the cornerstone of multiple studies (Rodriguez et al. (2015); Cao et al. (2015); İnkaya et al. (2015)).

In spite of the existence of these classic approaches, the design and implementation of novel meta-heuristics for addressing optimization and routing problems is a hot topic for the scientific community today. For this reason, many different metaheuristics have been proposed in the last decade, which have been successfully applied to various problems and fields. Some examples of these techniques are the artificial bee colony, proposed in 2005 by Karaboga (Karaboga (2005); Karaboga & Basturk (2007); Imanian et al. (2014); Moayedikia et al. (2015)), the imperialist competitive algorithm, presented by Gargari and Lucas in 2007 (Atashpaz-Gargari & Lucas (2007)), or the firefly algorithm, proposed by Yang in 2009 (Yang (2009)).

For this reason, this paper is focused on one metaheuristic proposed few years ago, called Bat Algorithm (BA). This population technique was proposed by Yang in 2010 (Yang (2010)), and it is based on the echolocation behavior of microbats, which can find their prey and discriminate different kinds of insects even in complete darkness. As can be read in several surveys (Yang & He (2013); Parpinelli & Lopes (2011)), since its proposal the BA has been successfully applied to different optimization fields and problems. Additionally, the fact that many research works focused on BA are being currently published confirms that BA still attracts a lot of interest (Fister et al. (2015); Meng et al. (2015)). As we have mentioned, the BA has been applied to many different optimization problems, anyway, it has been rarely applied to any routing problem (Saji et al. (2014)). This lack of works and the growing interest in the BA by the scientific community has motivated this work.

In this work, we present a discrete BA for solving routing problems. Being one of the first times that BA addresses this sort of problems, two of the most studied routing problems have been used for the experimentation: the TSP and the Asymmetric TSP (ATSP). Besides this, we also present an improved version of the basic BA (IBA), which outperforms the basic versions significantly.

The main objective of this study is to prove that the IBA is a promising approximation method for the TSP and ATSP. To prove that, we compare the results obtained by the IBA with the ones obtained by two basic versions of the BA, and with the ones obtained by five different well-known metaheuristics: GA, evolutionary simulated annealing (ESA) (Yip & Pao (1995)), the Island based Distributed Genetic Algorithm (IDGA) (Alba & Troya (1999)), a Discrete Firefly Algorithm (DFA), and a Discrete Imperialist Competitive Algorithm (DICA). To perform this comparison, 37 different TSP-ATSP instances have been used in the experimentation carried out. Furthermore, with the objective of drawing rigorous and fair conclusions, in addition to the conventional comparison based on the typical descriptive statistics parameters (results average, standard deviation, best result, etc.), we also perform three statistical tests along the paper: the Student's *t*-test, the Holm's test, and the Friedman test.

The remainder of this paper is structured as follows. In the following section (Section 2), a brief literature related to the BA is presented. In Section 3 the basic aspects of the BA are detailed. In Section 4 a brief description of the TSP and ATSP is made. Then, our proposed discrete BA and IBA are described in Section 5. Additionally, the experimentation carried out is described in Section 6. Finally, conclusions and future work are explained in Section 7.

2. Related Work

As we have mentioned in the previous section, the BA is a population algorithm proposed in 2010 by Yang. The basic BA is based on the echolocation or bio-sonar characteristics of microbats, and its first version was proposed for solving continuous optimization problems. Since this first implementation, the BA has been applied in a wide range of fields. Some of these fields are the continuous optimization, in which some additional works have been published apart from to the original one, (Bora et al. (2012); Yang & Hossein Gandomi (2012)), combinatorial optimization (Marichelvam et al. (2013)), image processing (Zhang & Wang (2012)) and clustering problems (Komarasamy & Wahi (2012)).

Besides this, many variations of the basic BA have been proposed in the literature. One example is the Fuzzy Logic BA (FLBA), presented by Khan et al. in 2011 (Khan et al. (2011)), which introduces some fuzzy logic mechanisms in the basic structure of the BA. This first FLBA was proposed as method for ergonomic screening of office workplaces. Another example of FLBA can be seen in (Pérez et al. (2015)). In this work the authors present a FLBA for dynamical parameter adaption. Other example of BA variation is the chaotic BA (CBA). The first CBA, which uses Lévy flights and chaotic maps, was proposed by Lin et al. for parameter estimation in dynamic biological systems

(Lin et al. (2012)). Furthermore, in 2014 an improved CBA was presented by Abdel-Raouf et al. for solving integer programming problems (Abdel-Raouf et al. (2014)). In the same year, Gandomi and Yang proposed a CBA for robust global optimization (Gandomi & Yang (2014)). Two other examples of BA variants are the BA with mutation (Zhang & Wang (2012)) or the multi-objective BA (Yang (2011)).

Additionally, some hybrid techniques have been developed using the BA as one of the hybridized methods. In (Pan et al. (2015)), for example, a hybrid particle swarm optimization with BA was developed. This approach, presented by Pan et al. in 2015, was implemented to deal with numerical optimization problems. One year earlier, Nguyen et al. proposed a hybrid bat algorithm with artificial bee colony also for solving the same kind of problems (Nguyen et al. (2014)). On the other hand, Meng et al. presented in 2015 a hybrid BA with differential evolution strategy for addressing constrained optimization problems (Meng et al. (2015)). Furthermore, Ramawan et al. developed in 2014 a BA hybridized with an artificial neural network. This novel approach was implemented to predict the output power of grid-connected photovoltaic system. At last, in (Roeva & Fidanova (2013)) a method which combines the BA with sequential quadratic programming for facing the parameter identification of an E. coli fed-batch cultivation process model was presented.

In the present work, we develop a discrete version of the BA. Although its first version was designed for continuous problems, the BA has been modified many times in the literature with the intention of addressing discrete optimization problems. In (Nakamura et al. (2012)), for instance, we can find the first Binary Bat Algorithm (BBA) applied to feature selection problems. Another successful BBA was developed by Mirjalili et al. in 2014 for solving discrete optimization problems (Mirjalili et al. (2014)). A recently paper published by Fister et al. presents another discrete version of the BA for the correct planning of the sports training sessions (Fister et al. (2015)). Finally, in (Luo et al. (2014)) a discrete BA was developed by Luo et al. for addressing the optimal permutation flow shop scheduling problem, and in (Marichelvam et al. (2013)) another discrete version of the BA was proposed for solving hybrid flow shop scheduling problems.

In spite of this great amount of research studies, as we have mentioned in the introduction of this paper, the BA has been rarely applied to any routing problem. This lack of studies has been the main motivation that has driven the realization of this work. Anyway, the main novelty of the presented IBA is not only its application field. The technique developed in this work presents the originality of using the Hamming Distance function to measure the distance between two bats of the swarm. This approach has been used previously in other techniques applied to the TSP, proving its good performance (Zhou et al. (2014)), but it has been never used for any BA. In addition, according to the basic philosophy of BA, all the bats of the swarm perform their movements always in the same way. This strategy is not used in the proposed IBA, where the bats are endowed with certain "intelligence". In this way, bats employ different movement schemes depending on the point of the solution space in which they are located. This is the first time that this approach is used in the literature.

On the other hand, as can be read in several studies of the literature, since its formulation, the TSP is one of the standard test problems used in performance analysis of discrete optimization algorithms (Mahi et al. (2015)). Many classic techniques have been applied to the TSP in the last few decades, as genetic algorithms (Grefenstette et al. (1985); Larrañaga et al. (1999)), simulated annealing (Malek et al. (1989); Aarts et al. (1988)), or the tabu search (Fiechter (1994); Knox (1994); Gendreau et al. (1998)). Despite being classical techniques, these methods have been also applied to the TSP in many recent studies (Misevičius (2015); Wang (2014); Nagata & Soler (2012)). More recent but still well-known techniques have been also widely applied for the TSP, as the ant colony optimization (Dorigo & Gambardella (1997); Jun-man & Yi (2012)), the variable neighborhood search (Carrabs et al. (2007); Burke et al. (2001)), or the particle swarm optimization (Clerc (2004); Shi et al. (2007)). As has been mentioned with respect to the previous ones, these techniques are also being applied nowadays to the TSP (Yao (2014); Pang et al. (2015); Ariyasingha & Fernando (2015)).

Additionally, the TSP has also been used as benchmarking problem for bio-inspired techniques proposed in the last decade. Some of these recent developed meta-heuristics are the firefly algorithm (Jati et al. (2013); Li et al. (2015)), which is based on the social behavior of fireflies and the phenomenon of bioluminescent communication, the Cuckoo Search (Ouaarab et al. (2014)), which is inspired by the breeding behaviour of cuckoos, or the Imperialist Competitive Algorithm (Ardalan et al. (2015); Yousefikhoshbakht & Sedighpour (2013)), which is a socio-politically motivated global search strategy, based on the imperialist competition of the countries. Some other examples of these recent developed bio-inspired meta-heuristics which have been applied to the TSP are the artificial bee colony algorithm (Karaboga & Gorkemli (2011)), inspired by the intelligent behaviour of honey bee swarm, or the Honey Bees Mating Optimization (Marinakis et al. (2011)).

It is also worth mentioning the hybrid techniques developed and tested with the TSP, as the one presented in (Saenphon et al. (2014)), which combines the well-known ant colony optimization with the gradient search. On the other hand, in (Mahi et al. (2015)) a hybrid method based on a particle swarm optimization, ant colony optimization and 3-opt algorithm is presented and applied to the TSP. Continuing with this concept, in (Chen & Chien (2011)) a technique hybridizing a genetic simulated annealing, an ant colony system and a particle swarm optimization was implemented and used to solve the TSP. Finally, another interesting study is the presented in (Zhao et al. (2015)), in which a simulated annealing hybridized with local searches is introduced.

In this study, in order to prove that the proposed meta-heuristic is a promising one, its performance is compared with three classical techniques, GA, ESA and IDGA, and two recently proposed ones, DICA and DFA.

Finally, it is noteworthy that this small set of works listed in this section is only a small sample of all the related work. Due to the high number of related papers, to summarize all the relevant work may be a huge task. For this reason, if any reader wants more information regarding the possible applications of BA, we recommend the reading of the literature review paper presented in (Yang & He (2013)). On the other hand, for further information of the TSP and its solving techniques, the work presented in (Anbuudayasankar et al. (2014)) is recommended.

3. Bat Algorithm

As we have briefly mentioned in previous sections, the BA is a bio-inspired metaheuristic based on the echolocation system of bats. In nature, bats emit ultrasonic pulses to the surrounding environment with hunting and navigation purposes. After the emission of these pulses, bats listen to the echoes, and based on them they can locate themselves and also locate and identify obstacles and preys. Furthermore, each bat of the swarm is able to find the most “nutritious” areas performing an individual search, or moving towards a “nutritious” location previously found by the swarm.

The main idea of the BA is to imitate this echolocation system of the bats. Anyway, some idealized rules have to be taken into account in order to make a proper adaptation (Yang (2010)):

- All bats use echolocation to detect the distance, and they have one “magic ability” that allow them to difference between a prey and an obstacle.
- All bats fly randomly with a velocity v_i at position x_i with a fixed frequency f_{min}, varying wavelength λ and loudness A_i to search a prey. In this idealized rule, we assume that every bat can adjust in an automatic way the frequency (or wavelength) of the emitted pulses, and the rate of these pulses emission $r \in [0,1]$. This automatic adjustment depends on the proximity of the targeted prey.
- In real situations, the loudness of bats emissions can vary in many ways. Nonetheless, we assume that this loudness can vary from a large positive A_0 to a minimum constant value A_{min}.

In Algorithm 1 the pseudocode of the basic BA is shown. Taking a look to this algorithm we can see that lines 1-6 correspond to the initialization process. First, the objective function has to be defined, and the initial population has to be initialized. We assume that every bat of the population represents one possible solution to the addressed problem. Then, all the parameters related to each bat are initialized and defined. These parameters are the velocity v_i, frequency f_i, pulse rate r_i and loudness A_i.

After these initialization steps, the algorithm starts its main phase. For each generation, every bat of the swarm moves by updating its velocity and position. For this movement, the following equations are used:

$$f_i = f_{min} + (f_{min} - f_{max})\beta \tag{1}$$

$$v_i^t = v_i^{t-1} + [x_i^{t-1} - x_*]f_i \tag{2}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{3}$$

where the parameter β is a randomly generated number in the [0,1] interval. Additionally, x_* denotes the current best solution in the swarm, and v_i^t and x_i^t represent the velocity and position of a bat i at time step t. Finally, the results of Equation (1) is used to control the range and pace of bats movement. In addition, for the local search part, whether a solution is selected among the best ones, a new solution for each bat is generated using a random walk

$$x_{new} = x_{old} + \epsilon A^t \tag{4}$$

where ϵ is a randomly generated number within the interval [-1,1], and A^t is the average loudness of the swarm at time step t. Finally, the loudness A_i and the rate r_i of each bat have to be updated if the conditions shown in the line 14 of Algorithm 1 are met. This update is conducted as follows:

Algorithm 1: Pseudocode of the basic BA

```
1  Define the objective function $f(x)$;
2  Initialize the bat population $X = x_1, x_2, ..., x_n$;
3  for each bat $x_i$ in the population do
4      Initialize the pulse rate $r_i$, velocity $v_i$ and loudness $A_i$;
5      Define the pulse frequency $f_i$ at $x_i$;
6  end
7  repeat
8      for each bat $x_i$ in the population do
9          Generate new solutions through Equations 1, 2 and 3;
10         if $rand > r_i$ then
11             Select one solution among the best ones;
12             Generate a local solution around the best one;
13         end
14         if $rand < A_i$ and $f(x_i) < f(x_*)$ then
15             Accept the new solution;
16             Increase $r_i$ and reduce $A_i$;
17         end
18     end
19 until termination criterion not reached;
20 Rank the bats and return the current best bat of the population;
```

$$A_i^{t+1} = \alpha A_i^{t+1} \tag{5}$$

$$r_i^{t+1} = r_i^0[1 - \exp(-\gamma t)] \tag{6}$$

where α and γ are constants. Thereby, for any $0<\alpha<1$ and $\gamma>0$ we have

$$a_i^t \to 0, r_i^t \to r_i^0, \ as\ t \to \infty \tag{7}$$

In many studies of the literature, $\alpha = \gamma$ is used in order to simplify the implementation of the algorithm. Specifically, we use $\alpha = \gamma = 0.98$ in this work. We have chosen this value empirically using a [0.90, 0.99] range.

4. The Traveling Salesman Problem

The TSP and ATSP are two of the most well-known and widely studied problems throughout history in computer science and operations research. As many other combinatorial optimization and routing problems, both problems are considered NP-Hard. This is the main reason of their great scientific interest. Thereby, the TSP and all its variants are used in a great number of research works every year as benchmarking problems (Urrutia et al. (2015); Wang (2015)). The TSP and ATSP can be defined as a complete graph $G = (V, A)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of vertices 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 these nodes. Besides that, each arc has an

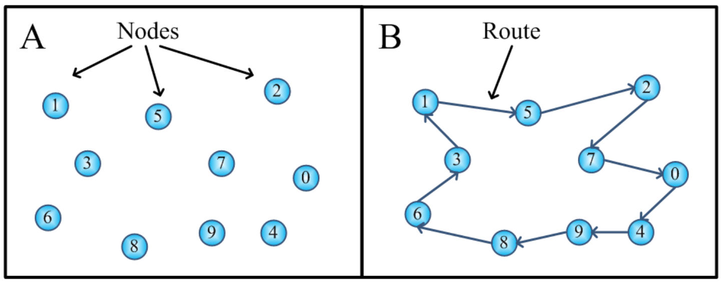

Figure 1: Possible TSP and ATSP 10-noded instance and a feasible solution

associated c_{ij} cost. In the symmetric version of the TSP the cost of traveling between two nodes is the same in both directions, i.e., $c_{ij} = c_{ji}$. On the other hand, although there may be arcs where $c_{ij} = c_{ji}$, in general $c_{ij} \neq c_{ji}$ for the ATSP.

The objective of the TSP and ATSP is to find a route that, starting and finishing at the same node, visits every node once and that minimizes the total cost of the path. The objective function for these problems is the total cost of the route.

In this paper, we have used the well-known path representation (Larranaga et al. (1999)) for the encoding of TSP and ATSP solutions. In this way, each solution is encoded as a permutation of numbers, which represents the order in which the nodes are visited. Using as an example a possible 10-node instance of the TSP, or ATSP, one solution would be represented as $X = (1, 5, 2, 7, 0, 4, 9, 8, 6, 3)$. This situation is depicted in Figure 1.

5. Our Improved Discrete Bat Algorithm for the TSP and ATSP

In this section, we will explain our adaption of the classic BA to solve the TSP and the ATSP (Section 5.1). Furthermore, we also explain an improved version of this algorithm (Section 5.2).

5.1. Discrete Bat Algorithm for the TSP and ATSP

First of all, it is noteworthy that, as has been said in Section 2, the original bat algorithm has been applied primarily to continuous optimization problems. As known, both TSP and ATSP are combinatorial optimization problems. Therefore, some modifications of the original BA are needed in order to prepare it for addressing the TSP and ATSP.

In our proposed algorithm, each bat in the swarm represents a possible and feasible solution for the TSP (or ATSP). Additionally, as has been detailed in Section 4, the total traveling cost of the route has been used as the objective function.

Regarding the basic parameters of the classic BA, which are r_i, A_i, f_i and v_i, the philosophy of the first two has remained in exactly the same form. In addition, in order simplify the complexity of the algorithm, the parameter "frequency", f_i has not been taken into account in our discrete versions of the BA. Finally, the "velocity", v_i, has been modified. In the basic version of the BA this parameter is calculated as has been shown in Equation (2):

$$v_i^t = v_i^{t-1} + [x_i^{t-1} - x_*]f_i$$

We can deduce from this formula that the velocity of a bat i at time step t depends on the v_i of the bat i in the previous time step, the difference between the bat i and the best bat in the swarm, and the f_i. As can be easily understood, this parameter cannot be used in the same way in our discrete version of the BA. With the intention of adapting the algorithm as accurately as possible, we have considered appropriate to relate v_i with the distance between the bat i and the best bat of the swarm. For this reason, we have adapted v_i using the well-known Hamming Distance in the following way:

$$v_i^t = \text{Random}[1, \text{HammingDistance}(x_i^t, x_*)] \tag{8}$$

In other words, the v_i of a bat i at time step t is a random number between 1, and the difference between this bat and the best bat of the swarm. This difference is represented by the Hamming Distance. The Hamming distance between two bats is the number of non-corresponding elements in the sequence. For example, taking into account the following two bats in a hypothetical TSP instance composed by 8 nodes,

$$x_1 : [0, 1, 2, 3, 4, 5, 6, 7]$$

$$x_2 : [0, 1, 3, 2, 5, 4, 6, 7]$$

the Hamming Distance between x_1 and x_2 would be 4.

Furthermore, regarding the generation of new solutions, in the classic BA the movement of the bats is made following the Equation 3:

$$x_i^t = x_i^{t-1} + v_i^t$$

In this case, we can deduce from this formula that the position of a bat i at time step t depends on the v_i of the bat i and its previous position at time step $t-1$. As previously said, this formula cannot be applied directly to the TSP and ATSP in this way. For this reason, we have developed a modification of it. In our study, two well-known successor operators have been used for the movement of the bats:

- *2-opt:* This function was defined by Lin in 1965 (Lin (1965)) and, since then, it has been widely used for solving routing problem (Tarantilis & Kiranoudis (2007); Bianchessi & Righini (2007)). The 2-opt eliminates at random two arcs within the existing path and creates two new arcs, avoiding the generation of sub tours.

- *3-opt:* The 3-opt operation, proposed also by Lin, is similar to 2-opt, with the difference that in this case the arcs removed are 3. The complexity of using this operator is greater than the 2-opt. Despite this, the operator has been used a large number of times throughout the history (Alfa et al. (1991); Rocki & Suda (2012)).

Thereby, the movement performed by each bat i at each time step t is the following:

$$x_i^t \leftarrow 2 - opt(x_i^{t-1}, v_i^t) \tag{9}$$

Namely, at each generation, each bat examines a v_i number of its neighbors, and it selects the best one as its current movement. In other words, the bat i performs a v_i number of 2-opt execution and it chooses the best one. In the case of 3-opt, Equation (9) becomes $x_i^t \leftarrow 3 - opt(x_i^{t-1}, v_i^t)$.

Finally, in relation to local procedure represented in lines 10-12 of Algorithm 1, if $rand > r_i$ one solution is selected among the best ones (in our experiments, one bat among the 10 best ones), and a local solution is generated around this one. To generate this local solution, the best neighbor of the chosen bat is selected using also the 2-opt and 3-opt moves.

5.2. Our proposed improvement for the discrete Bat Algorithm

The BA described in the previous section is the basic discrete BA that we have proposed for solving the TSP and ATSP. In addition, in this paper we also propose a simple but effective improvement in the structure of this basic BA. This improvement is related with the movement behavior of the bats. In the classic version of the BA, all the bats perform their movement following the same pattern throughout the entire execution, regardless of the point in the solution space in which each bat is located.

In our proposed IBA, we have provided some kind of intelligence to all the bats of the swarm. Thereby, each bat moves in a different way depending on its position in relation to the best bat of the swarm. In this way, when one bat is going to perform its movement, it examines its v_i^t. If this v_i^t is high (greater than $n/2$, where n is the number of nodes of the TSP-ATSP instance), we can assume that it is far from the best bat of the swarm, and we can conclude that it needs a *large move*. In the other case, if $v_i^t < n/2$, we can think that the bat is in a promising point of the solution space. Therefore, this bat will perform a *short move*. In our proposal, we have used the 2-opt for *short moves*, and the 3-opt as *large moves*.

This simple modification allows the population individuals to crawl the solution space using different neighborhood structures along the execution. This fact considerably enhances the exploration capacity of the technique, leading to an improvement in the results quality. Finally, the pseudocode of the proposed IBA is depicted in Algorithm 2

6. Experimentation

In this section the experimentation performed in this study is detailed. First of all, in Section 6.1, a qualitative comparison between the proposed IBA and the presented discrete BA is depicted. Then, in Section 6.2, the results obtained by the IBA are shown and compared with the ones obtained by the other five alternatives. Finally, in Section 6.3, the statistical analysis of these results and the convergence behaviour analysis are shown. All the tests conducted in this work have been performed on an Intel Core i5 2410 laptop, with 2.30 GHz and a RAM of 4 GB. Java has been used as the programming language. For the TSP, 22 instances have been used, and they have been obtained from the TSPLIB Benchmark (Reinelt (1991)). In addition, for the ATSP 15 instances have been chosen, obtained from the same benchmark. Overall, 37 instances have been used with 17 to 1002 nodes. Every instance has been run 20 times, and each one has a number in its name which represents the number of nodes it has.

Algorithm 2: Pseudocode of the proposed IBA. n = number of nodes of the instance.

```
1  Define the objective function $f(x)$;
2  Initialize the bat population $X = x_1, x_2, ..., x_n$;
3  for each bat $x_i$ in the population do
4      Initialize the pulse rate $r_i$, velocity $v_i$ and loudness $A_i$;
5  end
6  repeat
7      for each bat $x_i$ in the population do
8          Generate new solution;
9          if $v_i^t < n/2$ then
10             $x_i \leftarrow 2 - opt(x_i^{t-1}, v_i^t)$;
11         else
12             $x_i \leftarrow 3 - opt(x_i^{t-1}, v_i^t)$;
13         end
14         if $rand > r_i$ then
15             Select one solution among the best ones;
16             Generate a new bat selecting the best neighbor around the chosen bat
                 using the 2-opt or the 3-opt;
17         end
18         if $rand < A_i$ and $f(x_i) < f(x_*)$ then
19             Accept the new solution;
20             Increase $r_i$ and reduce $A_i$;
21         end
22     end
23 until termination criterion not reached;
24 Rank the bats and return the current best bat of the population;
```

6.1. Experimentation between presented discrete BA and the proposed IBA

As has been mentioned in the introduction of this section, an experimentation has been performed in order to prove that the proposed IBA performs better than the basic version of the BA. In this experimentation, the results obtained by the IBA for 35 TSP-ATSP instances have been compared with the ones obtained by two different versions of the basic BA. These results are shown in Table 2. In this Table, the results average, best solution found, standard deviation and average runtime (in seconds) are shown. In order to facilitate the replicability of this work, the parametrizations used for these three approaches are summarized in Table 1. It is important to highlight that the initial population of bats is randomly generated. In addition, as termination criterion, every execution finishes when there are $n + \sum_{k=1}^{n} k$ generations without improvements in the best solution, where n is the size of the problem.

In addition, in order to determine if IBA average is significantly different than the averages obtained by other techniques, we have performed Students t-test. The t statistic has the following form:

IBA		BA1		BA2	
Parameter	Value	Parameter	Value	Parameter	Value
Population size	50	Population size	50	Population size	50
Movement functions	2-opt & 3-opt	Movement function	2-opt	Movement function	3-opt
Initial A_i^0	Random number in [0.7,1.0]	Initial A_i^0	Random number in [0.7,1.0]	Initial A_i^0	Random number in [0.7,1.0]
Initial r_i^0	Random number in [0.0,0.4]	Initial r_i^0	Random number in [0.0,0.4]	Initial r_i^0	Random number in [0.0,0.4]
α & γ	0.98	α & γ	0.98	α & γ	0.98

Table 1: Parametrization of the IBA, BA1 and BA2 for the TSP and ATSP.

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

where:

$\overline{X_1}$: Average of IBA
SD_1: Standard deviation of IBA,
$\overline{X_2}$: Average of the other technique,
SD_2: Standard deviation of the other technique,
n_1: IBA size,
n_2: Size of the other technique,

In Table 2, we show a direct comparison between IBA and each of the other techniques using the Student's t-test. The t values shown can be positive, neutral, or negative. The double positive value (++) of t indicates that IBA is significantly better than the technique with which it is facing. In the opposite case (- -), IBA obtains significant worse solutions. If t is single positive (+), IBA shows to be better but not significantly. On the other hand, if the result is single negative (-), IBA demonstrates to be worse, but not in a significant way. Finally, a neutral value of t depicts equality in the results. We stated confidence interval at the 95% confidence level ($t_{0.05} = 1.96$). In this study the numerical value of t is also displayed. Thereby, the difference in results may be seen more easily.

As can be concluded viewing the results shown in Table 2, the IBA meets or outperforms the outcomes obtained by both basic discrete versions of the BA in the 100% of the cases. In addition, taking into account the performed Student's t-test, the differences in the results are significant in the 90% of the cases (63 out of 70 confrontations). The reason why the IBA is a better technique can be explained as follows: the bats that compose the population of the IBA have the option of exploring different neighborhood structures. This fact occurs because bats can switch their movement function method throughout the execution of the algorithm depending on the value of their v_i parameter. As we have explained in other studies (Osaba et al. (2014)), this change of neighborhood structure is an efficient mechanism to avoid local optima in routing problems, and it helps bats to explore the solution space in different ways.

6.2. Experimentation between the proposed IBA and the literature techniques

For sake of clarity, and before starting with the details of the performed experiments, we now briefly resume the basic principles of each technique which has been used in the experimentation.

Instance		Improved BA				Basic BA1					Basic BA2				
Name	Optima	Avg.	Best	S. dev.	Time	Avg.	Best	S. dev.	Time	t-test	Avg.	Best	S. dev.	Time	t-test
Oliver30	420	**420.0**	**420**	0.0	0.4	**420.0**	**420**	0.0	0.3	* (0.0)	**420.0**	**420**	0.0	0.4	* (0.0)
Eilon50	425	**427.4**	**425**	1.3	1.5	433.3	427	3.4	1.4	++ (7,2)	438.0	**425**	5.4	1.4	++ (8.5)
Eil51	426	**428.1**	**426**	1.6	1.7	438.3	430	2.5	1.7	++ (15.3)	436.8	429	5.3	1.9	++ (7.0)
Berlin52	7542	**7542.0**	**7542**	0.0	2.1	7676.0	**7542**	104.4	2.8	++ (5.7)	7681.9	**7542**	112.3	2.7	++ (5.5)
St70	675	**679.1**	**675**	2.8	3.9	696.4	**675**	6.3	4.1	++ (11.2)	694.3	**675**	9.7	4.6	++ (6.7)
Eilon75	535	**547.4**	**535**	3.9	4.5	555.6	545	7.3	4.7	++ (4.4)	562.7	550	8.9	5.1	++ (7.0)
Eil76	538	**548.1**	539	3.8	5.1	558.8	**538**	9.0	5.5	++ (4.8)	560.5	540	11.6	5.8	++ (4.5)
KroA100	21282	**21445.3**	**21282**	116.5	10.6	21884.2	21292	213.6	10.1	++ (8.0)	21989.4	21300	305.2	12.1	++ (7.4)
KroB100	22140	**22506.4**	**22140**	221.3	11.1	22842.9	22373	231.2	12.1	++ (4.7)	22946.7	22380	291.3	12.9	++ (5.3)
KroC100	20749	**21050.0**	**20749**	164.7	12.0	21476.6	20802	235.1	12.0	++ (6.6)	21631.1	20802	325.0	12.8	++ (7.1)
KroD100	21294	**21593.4**	**21294**	141.6	11.7	22001.4	21727	170.5	12.6	++ (8.2)	22053.5	21730	300.4	13.0	++ (6.1)
KroE100	22068	**22349.6**	**22068**	169.6	11.4	22771.5	22323	216.5	12.0	++ (6.8)	22790.2	22323	284.6	12.3	++ (5.9)
Eil101	629	**646.4**	634	4.9	13.1	667.1	640	4.4	13.5	++ (14.0)	670.0	642	8.4	14.1	++ (10.8)
Pr107	44303	**44793.8**	**44303**	232.4	12.1	45030.4	44618	184.4	14.4	++ (3.5)	45242.1	44701	259.3	15.8	++ (5.7)
Pr124	59030	**59412.1**	**59030**	265.9	18.5	59627.2	**59030**	395.9	19.7	+ (1.9)	59791.0	59074	448.3	20.5	++ (3.2)
Pr136	96772	**99351.2**	97547	707.2	23.4	101630.5	100485	732.8	24.2	++ (10.0)	101903.6	100500	893.1	25.3	++ (10.0)
Pr144	58537	**58876.2**	**58537**	295.6	30.3	58961.9	58588	227.4	29.9	+ (1.0)	59012.5	58602	301.0	31.2	+ (1.4)
Pr152	73682	**74676.9**	73921	426.5	31.0	74993.9	74172	429.3	28.5	++ (2.3)	75241.0	74172	539.3	30.0	++ (3.6)
Pr264	49135	**50908.3**	49756	887.0	92.5	52412.4	50256	995.3	90.3	++ (16.3)	52628.2	50306	1002.3	93.7	++ (18.5)
Pr299	48191	**49674.1**	48310	1200.1	147.2	50434.0	49142	1528.0	150.3	++ (5.6)	50232.6	49193	1739.1	153.2	++ (3.8)
br17	39	**39.0**	**39**	0.0	0.2	**39.0**	**39**	0.0	0.4	* (0.0)	**39.0**	**39**	0.0	0.8	* (0.0)
ftv33	1286	**1318.1**	**1286**	25.7	2.2	1390.6	1348	26.8	1.8	++ (8.7)	1387.3	1334	30.4	3.1	++ (7.7)
ftv35	1473	**1493.7**	**1473**	8.0	2.5	1559.6	1490	41.2	1.9	++ (7.0)	1571.9	1529	31.2	3.6	++ (10.8)
ftv38	1530	**1562.0**	**1530**	13.79	3.1	1603.2	**1530**	43.6	2.5	++ (4.0)	1657.8	1615	32.0	4.2	++ (12.2)
p43	5620	**5620.0**	**5620**	0.0	3.0	5654.6	5632	9.1	2.7	++ (16.9)	5621.2	**5620**	0.8	3.7	++ (6.0)
ftv44	1613	**1683.7**	**1613**	27.2	5.0	1774.5	1725	43.0	4.2	++ (7.9)	1817.4	1754	55.5	6.8	++ (9.6)
ftv47	1776	**1863.6**	1796	39.3	4.7	1959.3	1842	62.4	3.9	++ (5.8)	2031.6	1937	48.3	6.7	++ (12.0)
ry48p	14422	**14544.8**	**14422**	79.7	4.2	15172.9	14790	153.7	3.7	++ (16.2)	15069.7	14798	150.4	6.9	++ (13.7)
ft53	6905	**7294.1**	7001	196.9	6.5	7732.4	7105	252.6	5.1	++ (6.1)	7910.9	7452	151.8	8.2	++ (11.0)
ftv55	1608	**1737.5**	**1608**	50.5	6.9	1861.9	1686	81.1	5.3	++ (5.8)	1895.0	1806	45.7	8.3	++ (10.3)
ftv64	1839	**1999.2**	1879	68.2	7.2	2215.8	2068	74.8	6.2	++ (9.5)	2294.2	2122	58.1	9.0	++ (14.8)
ftv70	1950	**2233.2**	2111	48.8	8.1	2434.9	2238	84.3	6.7	++ (9.2)	2507.8	2314	110.8	10.5	++ (10.1)
ft70	38673	**40309.7**	39901	237.2	8.2	2434.9	2238	84.3	6.7	++ (6.3)	42506.3	42070	260.4	10.7	++ (27.8)
kro124p	36230	**39213.7**	37538	947.5	15.4	41772.6	40070	1872.2	13.2	++ (5.4)	43200.7	42307	611.7	18.7	++ (15.8)
rbg323	1326	**1640.9**	1615	30.4	243.6	1738.4	1713	16.5	237.4	++ (14.8)	1828.0	1713	92.7	251.7	++ (8.3)

Table 2: Results of the proposed IBA and two different Basic BAs for the TSP and ATSP.

Regarding the first of these techniques, the SA, it is one of the most popular local search techniques. It is based on the physical principle of cooling metal. Using that analogy, a SA generates an initial solution and the process proceeds by selecting new solutions randomly. This new solutions are not always better than the current ones, but they can be accepted probabilistically. Furthermore, as time passes and the temperature decreases (the metal becomes stronger), the probabilty of accepting worse solutions decreases, until it finally reaches 0. With the aim of performing a fair and rigorous experimentation, and taking into account that the IBA is a population technique, we have used a distributed version for the SA: the ESA (Yip & Pao (1995)).

On the other hand, GAs are one of the most successful meta-heuristics for solving combinatorial optimization problems. Thanks to their easy application and good performance, GAs have been used to solve many complex problems framed in various fields. GAs were proposed in 1975 by Holland (Holland (1975)), in an attempt to imitate the genetic process of living organisms, and the law of the evolution of species. Anyway, their practical use to solve complex optimization problems was shown later, by Goldberg (Goldberg (1989)) and De Jong (De Jong (1975)). With the aim of overcoming the drawbacks of GAs, such as premature convergence to a local optimum, and the

ESA		GA		IDGA	
Parameter	Value	Parameter	Value	Parameter	Value
Population size	50	Population size	50	Population size	4 subpob. of 13 individuals
Successor functions	2-opt & 3-opt	Crossover function	OX	Crossover functions	OX & OBX
Temperature	$-sup\Delta f/ln(p)$	Mutation functions	2-opt & 3-opt	Mutation functions	2-opt & 3-opt
Cooling constant	0.95	Cross. prob.	0.95	Cross. prob.	0.95, 0.9, 0.8 & 0.75
		Cross. prob.	0.25	Mut. prob.	0.05, 0.1, 0.2 & 0.25
		Selection func.	Binary tournament	Selection func.	Binary tournament
		Survivor func.	Binary tournament	Survivor func.	Binary tournament
				Migration Strat.	Best-Replace-Worst (Cantú-Paz (2001))

Table 3: Parametrization of the ESA, GA and IDGA for the TSP and ATSP. OX: Order Crossover (Davis (1985)). OBX: Order Based Crossover (Syswerda (1991)). $-sup\Delta f$ is the difference in the objective function of the best and the worse individuals of the initial population, and p=0.95.

imbalance between exploration and exploitation, Parallel GAs were proposed (PGA). PGAs are particularly easy to implement and promise substantial gains in performance. Reviewing the literature it can be seen that there are different ways to parallelize GA. The generally used classification divides parallel GAs in three categories: Fine Grain, Panmitic model and Island model. This last category is the most used, and it consists in a multiple populations that evolve separately most of the time and exchange individuals occasionally. This is the approach employed for the IDGA developed in our study.

Regarding the Imperialist Competitive Algorithm, it was proposed by Atashpaz-Gargari and Lucas (Atashpaz-Gargari & Lucas (2007)), and it is based on the concept of imperialism. The ICA divides the population into various empires, which evolve independently. Individuals of the population are called countries, and they are divided into two types: imperialist states (best country of the empire) and colonies. In this technique, the colonies make their movement through the solution space basing on the imperialist state. Meanwhile, empires compete between them, trying to conquer the weakest colonies of each other. This way, powerless empires could collapse and disappear, dividing their colonies among other empires.

The last used technique is a DFA. The first version of a Firefly Algorithm was proposed by Xin-She Yang in 2008. This nature-inspired algorithm is based on the flashing behaviour of fireflies, which acts as a signal system to attract other fireflies. As can be seen in several surveys (Fister et al. (2014, 2013)), the FA has been successfully applied to many different optimization fields and problems since its proposal. In addition, it still attracts a lot of interests in the current scientific community (Ma et al. (2015); Liang et al. (2015); Singh et al. (2015)).

As has been mentioned previous sections, the TSP is one of the standard test problems used in performance analysis of discrete optimization algorithms. In this way, even though the IBA obtains good results for both TSP and ATSP (it reaches the optimal solution in 14 out of 22 instance for the TSP and in 8 out of 15 for the ATSP, and in average, its solutions deviate a 1.38% from the optimal for the TSP and in 6,3% for the ATSP), it is important to highlight that the main goal of this study is not to find an optimal solution to these problems. Instead, we use both problems as benchmarking problems, which means that the principal objective of this research is to prove that the BA can be easily adapted to routing problems, and that the IBA is a promising approximation method for solving the TSP and ATSP. To reach this objective, we prove that the IBA can outperform some of the most used and well-known metaheuristics of the literature using classical non-heuristic functions. Thereby, for the results comparison we have chosen three of the most historically famous and successful techniques, the GA, the SA and the IDGA, and two recently proposed techniques, the DFA and the DICA.

It is important to highlight that, as far as possible, we have been used the same operators in similar parameters for all the algorithms implemented for the experimentation. In this way, our aim is to conclude which algorithm obtains better results using similar operators similar number of times. Furthermore, with the intention of facilitating the replicability of this study, we also show in Table 3 the parametrization used for these three algorithms. It is worth pointing out that all the individual are randomly generated. Besides this, we can find two successor functions for the ESA, which means that every individual has its own randomly assigned successor function. A similar procedure has been used in the IDGA with the crossover function, and in the GA and IDGA with the mutation function. Finally, as for the termination criterion, it is the same as for the IBA.

On the other hand, the parametrization used for IBA is the same shown in the previous section in Table 1. Additionally, the results obtained by the IBA, ESA, GA and IDGA techniques for the 37 instances are depicted in Table 4. In this table, we have shown the results average, best results, standard deviation and average runtime (in seconds). In addition, we have represented bolded the best results average, and the best solution found only whether it is the optimal one. As can be seen, and with the intention of not duplicating experiments, the results shown in Table 4 for the IBA are the same as have been depicted in Table 2.

The main conclusion that we can draw from this experimentation is that the IBA has been proved to be better than the ESA, GA and IDGA. Overall, the IBA has obtained better results in the 81.81% of the TSP instances (18 out of 22), being worse only in two instances (Eilon50 and Pr144). On the other hand, IBA has outperformed the other alternatives in the 73.33% of the ATSP instances, getting worse results only in two cases (ftv35 and rbg323). Technique by technique, the IBA has outperformed the ESA in 83.78% of the instances (31 out of 37). Furthermore, the IBA has performed better than the GA in 91.89% of the cases (34 out of 37) regarding the GA, and in 86.48% in relation to the IDGA (32 out of 37). Finally, it is important to highlight that the IBA has obtained worse results compared with each of the alternatives only in 4 occasions (in Pr144 and ftv35 with the ESA, and Eilon50 and rbg323 with the IDGA). This results are confirmed by the statistical tests shown in Section 6.3. Besides this, the IBA has reached the optimal solution in 59.45% of the instances (22 out of 37), outperforming the other techniques also in this respect.

Another important factor that is worth mentioning is the robustness of the IBA in relation to the other techniques. As can be seen in Table 4, the standard deviation of the results obtained by the IBA is lower than the ones presented by the other metaheuristics. This means that the quality of the solutions provided by the IBA move in a narrow range. This characteristic gives robustness and reliability to the algorithm, something that is very important if we want to use our technique in a real environment.

Finally, looking at the runtimes, we can say that the differences are not remarkable. While the IBA shows a slightly better performance in this respect compared with ESA and a similar behavior regarding the DGA, the GA has proved to be the best alternative. Anyway, as has been said, these differences are not remarkable and all the execution times shown by every technique are acceptable.

Instance		IBA				ESA				GA				IDGA			
Name	Optima	Avg.	Best	S. dev.	Time	Avg.	Best	S. dev.	Time	Avg.	Best	S. dev.	Time	Avg.	Best	S. dev.	Time
Oliver30	420	**420.0**	**420**	0.0	0.4	**420.0**	**420**	0.0	0.7	422.8	**420**	3.4	0.2	421.5	**420**	2.1	0.2
Eilon50	425	427.4	**425**	1.3	1.5	429.0	427	1.7	2.2	427.6	426	5.8	1.2	**427.0**	**425**	2.2	0.7
Eil51	426	**428.1**	**426**	1.6	1.7	431.6	**426**	2.9	2.1	440.8	427	7.3	1.7	434.4	**426**	4.5	1.2
Berlin52	7542	**7542.0**	**7542**	0.0	2.1	**7542.0**	**7542**	0.0	2.3	**7542.0**	**7542**	0.0	2.4	**7542.0**	**7542**	0.0	2.3
St70	675	**679.1**	**675**	2.8	3.9	682.1	**675**	3.9	4.5	709.8	**675**	5.7	4.2	690.2	**675**	9.8	4.1
Eilon75	535	**547.4**	**535**	3.9	4.5	550.2	545	3.9	5.4	565.6	550	14.2	5.6	552.4	544	7.6	4.4
Eil76	538	**548.1**	539	3.8	5.1	553.7	546	4.2	5.8	565.4	545	9.8	5.6	557.7	545	6.8	5.1
KroA100	21282	**21445.3**	**21282**	116.5	10.6	21481.7	**21282**	150.1	14.0	21812.4	21350	420.8	9.9	21731.8	21345	340.7	10.7
KroB100	22140	**22506.4**	**22140**	221.3	11.1	22602.2	22202	210.2	13.6	22687.4	22176	407.7	10.7	22712.6	22208	312.8	10.7
KroC100	20749	**21050.0**	**20749**	164.7	12.0	21170.4	**20749**	188.7	15.4	21510.4	20861	390.2	10.2	21298.7	20830	290.7	11.2
KroD100	21294	**21593.4**	**21294**	141.6	11.7	21726.5	21500	156.9	15.9	22184.6	21492	405.0	9.7	21696.9	21582	408.9	12.1
KroE100	22068	**22349.6**	**22068**	169.6	11.4	22499.7	22099	171.4	15.0	22741.3	22150	306.0	9.4	22721.9	22110	368.0	12.6
Eil101	629	**646.4**	634	4.9	13.1	658.4	650	4.4	16.3	673.8	655	12.5	10.6	660.7	650	7.5	11.7
Pr107	44303	**44793.8**	**44303**	232.4	12.1	44821.5	44413	179.3	16.7	45619.6	44392	1395.4	10.8	44902.5	44428	660.3	12.91
Pr124	59030	**59412.1**	**59030**	265.9	18.5	59593.6	**59030**	367.8	23.1	59901.0	**59030**	562.6	17.3	59912.8	59072	532.1	17.8
Pr136	96772	**99351.2**	97547	707.2	23.4	99858.3	98499	655.7	29.5	100472.4	98432	1225.6	23.8	99932.7	98532	1301.2	23.7
Pr144	58537	58876.2	**58537**	295.6	30.3	**58807.3**	58574	220.9	33.9	60591.4	58599	2342.8	32.8	58893.0	58581	1012.4	32.5
Pr152	73682	**74676.9**	73921	426.5	31.0	74969.5	74172	498.9	39.5	75658.3	74520	910.8	33.4	75126.7	74249	1005.7	32.0
Pr264	49135	**50908.3**	49756	887.0	92.5	52198.5	51603	426.1	102.5	52499.8	51712	932.4	92.1	52290.0	51653	782.7	94.5
Pr299	48191	**49674.1**	48310	1200.1	147.2	50532.3	49242	915.8	158.7	50817.1	49659	1585.7	147.6	50513.3	49572	1257.9	149.94
Pr439	107217	**115256.4**	11153	3825.8	201.9	116706.9	113497	4168.4	206.4	116943.4	113576	4642.4	208.4	116436.1	113207	4513.6	205.7
Pr1002	259047	**274419.7**	270016	3617.8	681.7	279419.7	273496	5120.3	683.1	279384.7	273001	5534.4	689.4	278951.4	272893	5617.4	687.1
br17	39	**39.0**	**39**	0.0	0.2	**39.0**	**39**	0.0	0.1	**39.0**	**39**	0.0	0.1	**39.0**	**39**	0.0	0.1
ftv33	1286	**1318.1**	**1286**	25.7	2.2	1322.5	**1286**	24.5	2.6	1409.4	1290	81.2	1.7	1402.7	**1286**	99.7	2.0
ftv35	1473	1493.7	**1473**	8.0	2.5	**1490.3**	**1473**	29.5	2.5	1597.2	1490	78.4	2.0	1589.0	1498	82.1	2.3
ftv38	1530	**1562.0**	**1530**	13.7	3.1	1568.8	**1530**	21.0	2.9	1670.4	1565	67.4	2.6	1650.1	1560	72.1	2.7
p43	5620	**5620.0**	**5620**	0.0	3.0	**5620.0**	**5620**	0.0	2.5	5625.2	5620	5.4	2.8	**5620.0**	5620	0.0	2.6
ftv44	1613	**1683.7**	**1613**	27.2	5.0	1718.9	1645	39.2	5.3	1780.0	1649	94.7	4.6	1800.3	1645	120.7	5.3
ftv47	1776	**1863.6**	1796	39.3	4.7	1879.8	1795	52.7	5.2	1963.1	1820	89.6	4.3	1957.4	1822	118.4	4.7
ry48p	14422	**14544.8**	**14422**	79.7	4.2	14598.0	14485	108.7	4.6	14992.1	14545	340.7	3.9	14892.0	14530	201.8	4.3
ft53	6905	**7294.1**	7001	196.9	6.5	7314.7	6990	157.8	6.6	7568.4	7270	358.7	6.2	7445.2	7076	430.0	6.1
ftv55	1608	**1737.5**	**1608**	50.5	6.9	1822.6	1725	70.1	7.2	1871.1	1700	132.1	5.8	1970.8	1842	120.1	6.7
ftv64	1839	**1999.2**	1879	68.2	7.2	2072.3	1955	65.0	7.1	2205.7	2014	127.4	6.9	2262.1	2080	152.1	6.8
ftv70	1950	**2233.2**	2111	48.8	8.1	2312.6	2200	67.2	8.0	2315.8	2184	140.7	7.9	2351.7	2135	134.2	7.5
ft70	38673	**40309.7**	39901	237.2	8.2	40551.4	39650	467.2	8.7	40400.7	39407	620.4	7.6	40672.4	39241	781.8	8.2
kro124p	36230	**39213.7**	37538	947.5	15.4	42132.0	40019	1250.7	17.7	42250.3	39265	1825.4	15.3	42101.9	39099	1072.4	15.8
rbg323	1326	1640.9	1615	30.4	243.6	1685.0	1620	72.1	246.8	1631.5	1514	77.2	238.1	**1623.5**	1510	82.2	238.7

Table 4: Results of the proposed IBA and ESA, GA and IDGA for the TSP and ATSP.

On the other hand, in Table 5 the results obtained by the IBA in the 37 instances are compared with the ones obtained by a DFA and a DICA. In order to implement both these algorithms, the guidelines given in (Li et al. (2015)) and (Yousefikhoshbakht & Sedighpour (2013)) have been followed. As we have done previously, we have used similar parameters and functions in these techniques with the intention of obtaining fair conclusions. In this way, the same number of individuals has been used for the IBA, DFA and DICA, and all these techniques base the movements of these individuals on the Hamming Distance function.

The conclusions that can be obtained from this second table are similar than the ones drawn previously. In this case, the IBA is the technique with the better performance, obtaining better results in the 67.56% of the instances (25 out of 37), being worse in 8 cases. Overall, IBA has outperformed DFA and DICA in the 72.72% of the TSP cases, and in the 60% of the ATSP instances, getting worse results in 4 cases of each problem. Technique by technique, the IBA has performed better than the DFA in the 70.27% of the cases (26 out of 37). Additionally, the IBA has outperformed the DICA in the 83.78% of the instances (31 out of 37). Besides this, as in the comparison with the other

Instance		IBA				DFA				DICA			
Name	Optima	Avg.	Best	S. dev.	Time	Avg.	Best	S. dev.	Time	Avg.	Best	S. dev.	Time
Oliver30	420	**420.0**	**420**	0.0	0.4	**420.0**	**420**	0.0	0.4	**420.**0	**420**	0.0	0.5
Eilon50	425	427.4	**425**	1.3	1.5	**427.2**	**425**	1.8	1.3	427.9	**425**	2.1	1.4
Eil51	426	**428.1**	**426**	1.6	1.7	430.8	**426**	2.3	1.6	432.3	426	3.1	1.8
Berlin52	7542	**7542.0**	**7542**	0.0	2.1	**7542.0**	**7542**	0.0	2.2	**7542.0**	**7542**	0.0	2.5
St70	675	**679.1**	**675**	2.8	3.9	685.3	**675**	4.0	4.3	684.7	**675**	3.7	4.1
Eilon75	535	547.4	**535**	3.9	4.5	**543.6**	**535**	5.3	4.6	551.7	537	6.8	5.6
Eil76	538	**548.1**	539	3.8	5.1	556.8	543	4.9	5.3	557.6	544	5.8	5.3
KroA100	21282	**21445.3**	**21282**	116.5	10.6	21483.6	**21282**	163.7	10.3	21500.3	21282	183.4	10.8
KroB100	22140	**22506.4**	**22140**	221.3	11.1	22604.8	22183	243.9	11.6	22599.7	22180	244.9	11.3
KroC100	20749	**21050.0**	**20749**	164.7	12.0	21096.3	20756	148.3	12.8	21103.9	20756	161.1	11.7
KroD100	21294	**21593.4**	**21294**	141.6	11.7	21683.8	21408	163.7	12.4	21666.8	21399	174.0	12.6
KroE100	22068	**22349.6**	**22068**	169.6	11.4	22413.0	22079	183.0	11.6	22453.3	22083	196.9	11.7
Eil101	629	**646.4**	634	4.9	13.1	659.0	643	8.1	13.3	663.8	644	9.6	12.0
Pr107	44303	44793.8	**44303**	232.4	12.1	**44790.4**	**44303**	227.3	12.6	44803.3	**44303**	302.7	12.9
Pr124	59030	59412.1	**59030**	265.9	18.5	**59404.3**	**59030**	257.9	18.8	59436.9	**59030**	299.4	19.0
Pr136	96772	**99351.2**	97547	707.2	23.4	99683.7	97716	831.3	24.1	99583.7	97736	848.9	24.0
Pr144	58537	**58876.2**	**58537**	295.6	30.3	58993.3	58546	300.1	30.9	59070.9	58563	323.0	30.7
Pr152	73682	**74676.9**	73921	426.5	31.0	74934.3	74033	483.7	32.1	74886.7	74052	513.9	32.0
Pr264	49135	**50908.3**	49756	887.0	92.5	51837.0	50491	760.8	93.0	51943.6	50553	863.7	94.1
Pr299	48191	**49674.1**	48310	1200.1	147.2	49839.7	48579	1305.4	149.1	49880.3	48600	1413.7	150.3
Pr439	107217	**115256.4**	111538	3825.8	201.9	115558.2	111967	4009.1	202.4	115763.1	111983	4219.6	203.7
Pr1002	259047	**274419.7**	270016	3617.8	681.7	277344.7	272003	4731.6	682.0	277308.1	272082	4293.7	684.6
br17	39	**39.0**	**39**	0.0	0.2	**39.0**	**39**	0.0	0.2	**39.0**	**39**	0.0	0.3
ftv33	1286	**1318.1**	**1286**	25.7	2.2	1320.9	**1286**	21.9	2.8	1324.6	**1286**	28.3	2.9
ftv35	1473	1493.7	**1473**	8.0	2.5	1498.8	**1473**	10.4	2.7	**1490.6**	**1473**	11.9	2.8
ftv38	1530	1562.0	**1530**	13.7	3.1	**1560.4**	**1530**	14.6	3.0	1565.6	1530	15.8	3.2
p43	5620	**5620.0**	**5620**	0.0	3.0	**5620.0**	**5620**	0.0	2.8	**5620.0**	**5620**	0.0	3.1
ftv44	1613	**1683.7**	**1613**	27.2	5.0	1690.8	1620	32.3	5.1	1694.3	1622	54.0	5.3
ftv47	1776	1863.6	1796	39.3	4.7	**1858.3**	1795	63.4	5.5	1873.0	1799	70.1	5.8
ry48p	14422	**14544.8**	**14422**	79.7	4.2	14694.4	14453	94.7	4.4	14689.8	14463	73.6	5.4
ft53	6905	**7294.1**	7001	196.9	6.5	7302.0	6993	186.4	6.8	7320.1	7002	200.3	6.9
ftv55	1608	**1737.5**	**1608**	50.5	6.9	1790.6	1628	64.0	7.0	1801.4	1630	83.0	7.2
ftv64	1839	**1999.2**	1879	68.2	7.2	2041.6	1903	73.4	7.0	2040.8	1900	81.3	7.3
ftv70	1950	**2233.2**	2111	48.8	8.1	2290.8	2173	60.0	7.8	2322.6	2167	63.3	8.3
ft70	38673	**40309.7**	39901	237.2	8.2	40694.8	39668	494.6	8.5	40699.7	39660	534.9	8.8
kro124p	36230	**39213.7**	37538	947.5	15.4	41637.5	39438	1094.7	15.8	41608.3	39400	1116.8	15.7
rbg323	1326	1640.9	1615	30.4	243.6	**1634.7**	1599	34.6	245.1	1639.7	1600	31.1	247.0

Table 5: Results of the proposed IBA, DFA and DICA for the TSP and ATSP.

techniques, the IBA has reached the optimal solution in more occasions that the DFA and DICA, outperforming these techniques also in this respect. Finally, the reflections about the robustness and the runtimes are the same as in the previous analysis.

6.3. Statistical analysis and convergence behaviour analysis

Two different statistical tests have been conducted with the results obtained in previous subsections in order to obtain rigorous and fair conclusions. The guidelines given by Derrac et al. in (Derrac et al. (2011)) have been followed to perform this statistical analysis. First of all, the Friedman's non-parametric test for multiple comparisons has been used to check if there are any significant differences among all the techniques. For the TSP problem (Table 6), the resulting Friedman statistic has been 65.525. Taking into account that the confidence interval has been stated at the 99% confidence level, the critical point in a χ^2 distribution with 5 degrees of freedom is 15.086. Since 65.525>15.086, it can be concluded that there are significant differences among the results reported by the five compared algorithms, being IBA the one with the lowest rank. Finally, regarding this Friedman's test, the computed p-value has been 0.0.

On the other hand, the resulting Friedman statistic for the ATSP has been 27.761. Because 27.761>15.086, it can be concluded also for the ATSP that there are significant differences, being the IBA the best technique. In this case, the computed p-value has been 0.000041.

To evaluate the statistical significance of the better performance of IBA, the Holm's post-hoc test has been conducted using IBA as control algorithm. The unadjusted and adjusted p-values obtained through the application of Holm's post-hoc procedure can be seen in Table 7. Analyzing this data, and taking into account that all the p-values are lower than 0.05, it can be concluded that IBA is significantly better for the TSP at a 95% confidence level. On the other hand, for the ATSP, the IBA is significantly better than the GA, IDGA, ESA and DICA, and better, but not significantly, than the DFA.

TSP		ATSP	
Algorithm	Ranking	Algorithm	Ranking
IBA	1.4545	IBA	1.8333
ESA	3.5455	ESA	3.5333
GA	5.5682	GA	4.9667
IDGA	4.5455	IDGA	4.5
DFA	2.5909	DFA	2.7667
DICA	3.2955	DICA	3.4

Table 6: Average rankings returned by the Friedman's non-parametric test for the TSP and ATSP problems.

TSP			ATSP		
Algorithm	Unadjusted p	Adjusted p	Algorithm	Unadjusted p	Adjusted p
GA	0	0	GA	0.000005	0.000023
IDGA	0	0	IDGA	0.000095	0000379
ESA	0.00021	0.00063	ESA	0.012827	0.0.03848
DICA	0.0011	0.0022	DICA	0.021827	0.043654
DFA	0.043951	0.043951	DFA	0.171857	0.171857

Table 7: Unadjusted and adjusted p-values obtained for the TSP and ATSP through the application of Holm's post-hoc procedure using IBA as control algorithm.

To conclude the whole analysis of the results, and with the aim of making a deeper analysis, the convergence behavior shown by the IBA is compared next with the ones shown by the ESA and the DFA. We have selected the ESA and the DFA for this comparison because they are the most similar techniques in terms of average results quality with respect to IBA. In Table 8, the average number of objective function evaluations needed to reach the final solution for each instance is shown (in thousands), as well as the standard deviations.

Analyzing the results shown in Table 8, we can conclude that IBA outperforms both ESA and DFA also in terms of convergence. Overall, the IBA has shown a better performance in the 62.85%of the instances (22 out of 35). Specifically, it can be concluded that the IBA performs better than the ESA in instances with, approximately, 100 nodes or less. This behavior has not been shown in the case of DFA, where the IBA has proved to be better in general terms. This fact provides an advantage to the IBA, since it can obtain better results needing less number of objective function evaluations.

As a final conclusion, we can say that using similar functions and parameters, the proposed IBA outperforms the other alternatives needing similar and acceptable runtimes and showing a better robustness and convergence. In addition, the improvements shown

Instance	IBA		ESA		DFA	
Name	Avg.	S. dev.	Avg.	S. dev.	Avg.	S. dev.
Oliver30	**2.17**	0.48	23.91	3.49	3.38	1.45
Eilon50	22.8	8.42	94.37	32.31	**17.66**	8.68
Eil51	**15.37**	14.13	85.91	22.60	17.56	6.93
Berlin52	**20.07**	6.02	128.26	49.82	23.68	7.31
St70	72.67	30.38	216.08	77.49	**69.56**	32.82
Eilon75	**116.56**	65.21	273.23	89.97	173.50	83.06
Eil76	**91.53**	30.89	262.89	81.37	164.18	69.60
KroA100	**739.86**	177.58	784.84	192.21	812.56	126.93
KroB100	**461.05**	159.51	729.83	260.82	813.68	127.31
KroC100	872.51	199.52	**726.35**	228.34	835.79	145.82
KroD100	**600.31**	220.75	689.49	230.11	875.74	234.70
KroE100	**602.94**	103.73	791.76	219.53	843.72	197.93
Eil101	**512.73**	122.32	598.11	120.32	617.83	148.16
Pr107	679.07	118.14	**661.97**	135.71	713.90	206.01
Pr124	1602.51	436.32	**1446.91**	345.61	1589.71	399.07
Pr136	2866.60	927.02	**2318.20**	655.24	2763.80	883.35
Pr144	4361.11	1421.23	**3678.46**	943.12	4097.09	1349.64
Pr152	4853.19	1639.27	**3853.91**	1037.53	4769.37	1709.60
Pr264	6375.46	1864.76	**6096.45**	1749.12	6686.39	2009.73
Pr299	**6597.94**	2001.91	6731.23	2067.71	7016.91	2364.28
Pr439	8346.85	2739.64	**8006.91**	2996.35	8736.28	3069.18
Pr1002	12103.73	4964.3	**11038.32**	4722.71	12843.60	5207.21
br17	0.31	0.04	5.65	0.63	**0.29**	0.03
ftv33	14.66	9.43	52.85	11.75	**13.98**	7.81
ftv35	**12.78**	4.42	50.80	14.90	14.12	5.16
ftv38	25.53	9.68	49.89	13.30	**23.43**	10.51
p43	**11.87**	4.15	50.81	9.12	13.80	5.05
ftv44	**41.84**	21.27	81.54	38.03	45.04	20.65
ftv47	47.81	24.32	83.71	32.03	**46.65**	26.86
ry48p	**42.11**	13.32	79.45	19.52	49.16	18.70
ft53	**61.68**	30.66	99.30	39.52	65.25	29.23
ftv55	**71.81**	28.74	124.85	53.85	75.19	32.29
ftv64	**89.86**	40.15	143.89	65.42	93.40	42.95
ftv70	**134.98**	50.30	170.91	73.23	146.76	57.61
ft70	**131.59**	56.91	180.94	77.21	136.95	63.29
kro124p	**438.59**	101.62	641.49	163.99	451.02	134.79
rbg323	**6948.06**	1393.96	7316.76	2031.78	7006.54	1681.71

Table 8: Convergence of IBA, ESA and DFA for TSP and ATSP, expressed in thousand of objetive function evaluations

are significant in most cases. For this reason, we can say that the presented IBA is a promising approximation method to solve the TSP and ATSP, meeting, in this respect, the main objective of this study.

7. Conclusions and Further Work

In this work we have presented the first Discrete Bat Algorithm for solving the Traveling Salesman Problem and the Asymmetric Traveling Salesman Problem. In addition, we have proposed an improved version of the basic BA. In this Improved Bat Algorithm, bats are endowed with some kind of "intelligence". This intelligence makes the bats follow different patterns of movement depending on the point of the solution space in which they are located. In order to prove that the proposed IBA is a promising approximation method to solve the TSP and ATSP, we have compared its performance along 37 instances with the one of two basic BAs. Furthermore, the results obtained by the IBA have been compared with those obtained by five different metaheuristics:

a genetic algorithm, an evolutionary simulated annealing an island based distributed genetic algorithm, a discrete firefly algorithm and a discrete imperialist competitive algorithm. Additionally, three statistical tests have been conducted along the paper with the obtained outcomes: the Student's t-test, Holm's test and the Friedman test. Overall, the IBA has demonstrated a great performance for the TSP and ATSP, outperforming all the other alternatives, being the improvements significant in most of the cases.

Both TSP and ATSP problems are standard discrete problems; however, the conclusions obtained in this study cannot be generalized to other discrete problems. For this reason, as a future work, we intend to develop some additional versions of the proposed IBA to solve other routing problems. Our planned work includes the application of the IBA for the Capacitated Vehicle Routing Problem (Ralphs et al. (2003)), and more complex routing problems, such as Rich Vehicle Routing Problems (Caceres-Cruz et al. (2014)). Additionally, we are aware of the large number of existing meta-heuristics in the literature. The comparison of the IBA with the five selected techniques is enough to prove that the proposed metaheuristic is a promising one. Nevertheless, we think that a wider experimentation with additional techniques can be valuable for the scientific community. In addition, we are planning to compare the proposed technique with exact methods and commercial solvers, using metrics such as the computational time or the number of explored solutions. These techniques are not similar to the IBA in terms of concepts and philosophy. However, we think such comparison will be very useful and can provide some insight into these methods.

Acknowledgement

This project was supported by the European Unions Horizon 2020 research and innovation programme through the TIMON: Enhanced real time services for optimized multimodal mobility relying on cooperative networks and open data project (636220). As well as by the projects TEC2013-45585-C2-2-R from the Spanish Ministry of Economy and Competitiveness, and PC2013-71A from the Basque Government.

Aarts, E. H., Korst, J. H., & van Laarhoven, P. J. (1988). A quantitative analysis of the simulated annealing algorithm: A case study for the traveling salesman problem. *Journal of Statistical Physics*, *50*, 187–206.

Abdel-Raouf, O., Abdel-Baset, M., & El-Henawy, I. (2014). An improved chaotic bat algorithm for solving integer programming problems. *International Journal of Modern Education and Computer Science*, *6*, 18.

Alba, E., & Troya, J. M. (1999). A survey of parallel distributed genetic algorithms. *Complexity*, *4*, 31–52.

Alfa, A., Heragu, S., & Chen, M. (1991). A 3-opt based simulated annealing algorithm for vehicle routing problems. *Computers & Industrial Engineering*, *21*, 635–639.

Anbuudayasankar, S., Ganesh, K., & Mohapatra, S. (2014). Survey of methodologies for tsp and vrp. In *Models for Practical Routing Problems in Logistics* (pp. 11–42). Springer.

Ardalan, Z., Karimi, S., Poursabzi, O., & Naderi, B. (2015). A novel imperialist competitive algorithm for generalized traveling salesman problems. *Applied Soft Computing*, *26*, 546–555.

Ariyasingha, I., & Fernando, T. (2015). Performance analysis of the multi-objective ant colony optimization algorithms for the traveling salesman problem. *Swarm and Evolutionary Computation*, .

Atashpaz-Gargari, E., & Lucas, C. (2007). Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In *IEEE Congress on Evolutionary Computation* (pp. 4661–4667).

Bezerra, L. C., López-Ibáñez, M., & Stützle, T. (2014). Automatic design of evolutionary algorithms for multi-objective combinatorial optimization. In *Parallel Problem Solving from Nature* (pp. 508–517). Springer.

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

Bora, T. C., Coelho, L. d. S., & Lebensztajn, L. (2012). Bat-inspired optimization approach for the brushless dc wheel motor problem. *IEEE Transactions on Magnetics*, *48*, 947–950.

Bortfeldt, A., Hahn, T., Männel, D., & Mönch, L. (2015). Hybrid algorithms for the vehicle routing problem with clustered backhauls and 3d loading constraints. *European Journal of Operational Research*, *243*, 82–96.

Burke, E. K., Cowling, P. I., & Keuthen, R. (2001). Effective local and guided variable neighbourhood search methods for the asymmetric travelling salesman problem. In *Applications of Evolutionary Computing* (pp. 203–212). Springer.

Caceres-Cruz, J., Arias, P., Guimarans, D., Riera, D., & Juan, A. A. (2014). Rich vehicle routing problem: Survey. *ACM Computing Surveys*, *47*, 32.

Cantú-Paz, E. (2001). Migration policies, selection pressure, and parallel evolutionary algorithms. *Journal of heuristics*, *7*, 311–334.

Cao, B., Glover, F., & Rego, C. (2015). A tabu search algorithm for cohesive clustering problems. *Journal of Heuristics*, (pp. 1–21).

Carrabs, F., Cordeau, J.-F., & Laporte, G. (2007). Variable neighborhood search for the pickup and delivery traveling salesman problem with lifo loading. *INFORMS Journal on Computing*, *19*, 618–632.

Chen, S.-M., & Chien, C.-Y. (2011). Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques. *Expert Systems with Applications*, *38*, 14439–14450.

Christofides, N. (1976). The vehicle routing problem. *RAIRO-Operations Research-Recherche Opérationnelle*, *10*, 55–70.

Clerc, M. (2004). Discrete particle swarm optimization, illustrated by the traveling salesman problem. In *New optimization techniques in engineering* (pp. 219–239). Springer.

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

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

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, *1*, 3–18.

Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical computer science*, *344*, 243–278.

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

Fiechter, C.-N. (1994). A parallel tabu search algorithm for large traveling salesman problems. *Discrete Applied Mathematics*, *51*, 243–267.

Fister, I., Fister Jr, I., Yang, X.-S., & Brest, J. (2013). A comprehensive review of firefly algorithms. *Swarm and Evolutionary Computation*, .

Fister, I., Rauter, S., Yang, X.-S., & Ljubič, K. (2015). Planning the sports training sessions with the bat algorithm. *Neurocomputing*, *149*, 993–1002.

Fister, I., Yang, X.-S., Fister, D., & Fister Jr, I. (2014). Firefly algorithm: A brief review of the expanding literature. In *Cuckoo Search and Firefly Algorithm* (pp. 347–360). Springer.

Gandomi, A. H., & Yang, X.-S. (2014). Chaotic bat algorithm. *Journal of Computational Science*, *5*, 224–232.

Gendreau, M., Laporte, G., & Semet, F. (1998). A tabu search heuristic for the undirected selective travelling salesman problem. *European Journal of Operational Research*, *106*, 539–545.

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

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

Grefenstette, J., Gopal, R., Rosmaita, B., & Van Gucht, D. (1985). Genetic algorithms for the traveling salesman problem. In *Proceedings of the first International Conference on Genetic Algorithms and their Applications* (pp. 160–168). Lawrence Erlbaum, New Jersey (160-168).

Groba, C., Sartal, A., & Vázquez, X. H. (2015). Solving the dynamic traveling salesman problem using a genetic algorithm with trajectory prediction: An application to fish aggregating devices. *Computers*

& Operations Research, *56*, 22–32.

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

Imanian, N., Shiri, M. E., & Moradi, P. (2014). Velocity based artificial bee colony algorithm for high dimensional continuous optimization problems. *Engineering Applications of Artificial Intelligence*, *36*, 148–163.

İnkaya, T., Kayalıgil, S., & Özdemirel, N. E. (2015). Ant colony optimization based clustering methodology. *Applied Soft Computing*, *28*, 301–311.

Jati, G. K., Manurung, R., & Suyanto (2013). Discrete firefly algorithm for traveling salesman problem: A new movement scheme. *Swarm Intelligence and Bio-Inspired Computation:Theory and Applications*, (pp. 295–312).

Jun-man, K., & Yi, Z. (2012). Application of an improved ant colony optimization on generalized traveling salesman problem. *Energy Procedia*, *17*, 319–325.

Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization*. Technical Report Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of global optimization*, *39*, 459–471.

Karaboga, D., & Gorkemli, B. (2011). A combinatorial artificial bee colony algorithm for traveling salesman problem. In *International Symposium on Innovations in Intelligent Systems and Applications* (pp. 50–53). IEEE.

Kasperski, A., & Zieliński, P. (2015). Combinatorial optimization problems with uncertain costs and the owa criterion. *Theoretical Computer Science*, *565*, 102–112.

Kennedy, J., Eberhart, R. et al. (1995). Particle swarm optimization. In *Proceedings of IEEE international conference on neural networks* (pp. 1942–1948). Perth, Australia volume 4.

Khan, K., Nikov, A., & Sahai, A. (2011). A fuzzy bat clustering method for ergonomic screening of office workplaces. In *Third International Conference on Software, Services and Semantic Technologies* (pp. 59–66). Springer.

Kirkpatrick, S., Gellat, C., & Vecchi, M. (1983). Optimization by simmulated annealing. *science*, *220*, 671–680.

Knox, J. (1994). Tabu search performance on the symmetric traveling salesman problem. *Computers & Operations Research*, *21*, 867–876.

Komarasamy, G., & Wahi, A. (2012). An optimized k-means clustering technique using bat algorithm. *European Journal of Scientific Research*, *84*, 26–273.

Laporte, G. (1992a). The traveling salesman problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, *59*, 231–247.

Laporte, G. (1992b). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, *59*, 345–358.

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

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

Lawler, E. L., Lenstra, J. K., Kan, A. R., & Shmoys, D. B. (1985). *The traveling salesman problem: a guided tour of combinatorial optimization* volume 3. Wiley New York.

Levin, M. S. (2015). System configuration and combinatorial optimization. In *Modular System Design and Evaluation* (pp. 89–109). Springer.

Li, M., Ma, J., Zhang, Y., Zhou, H., & Liu, J. (2015). Firefly algorithm solving multiple traveling salesman problem. *Journal of Computational and Theoretical Nanoscience*, *12*, 1277–1281.

Liang, R.-H., Wang, J.-C., Chen, Y.-T., & Tseng, W.-T. (2015). An enhanced firefly algorithm to multi-objective optimal active/reactive power dispatch with uncertainties consideration. *International Journal of Electrical Power & Energy Systems*, *64*, 1088–1097.

Lin, J.-H., Chou, C.-W., Yang, C.-H., Tsai, H.-L. et al. (2012). A chaotic levy flight bat algorithm for parameter estimation in nonlinear dynamic biological systems. *Computer and Information Technology*, *2*, 56–63.

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

Luo, Q., Zhou, Y., Xie, J., Ma, M., & Li, L. (2014). Discrete bat algorithm for optimal problem of permutation flow shop scheduling. *The Scientific World Journal*, *2014*.

Ma, Y., Zhao, Y., Wu, L., He, Y., & Yang, X.-S. (2015). Navigability analysis of magnetic map with projecting pursuit-based selection method by using firefly algorithm. *Neurocomputing*, .

Mahi, M., Baykan, Ö. K., & Kodaz, H. (2015). A new hybrid method based on particle swarm optimization, ant colony optimization and 3-opt algorithms for traveling salesman problem. *Applied Soft Computing*, *30*, 484–490.

Malek, M., Guruswamy, M., Pandya, M., & Owens, H. (1989). Serial and parallel simulated annealing and tabu search algorithms for the traveling salesman problem. *Annals of Operations Research*, *21*, 59–84.

Marichelvam, M., Prabaharan, T., Yang, X.-S., & Geetha, M. (2013). Solving hybrid flow shop scheduling problems using bat algorithm. *International Journal of Logistics Economics and Globalisation*, *5*, 15–29.

Marinakis, Y., Marinaki, M., & Dounias, G. (2011). Honey bees mating optimization algorithm for the euclidean traveling salesman problem. *Information Sciences*, *181*, 4684–4698.

Meng, X., Gao, X., & Liu, Y. (2015). A novel hybrid bat algorithm with differential evolution strategy for constrained optimization. *International Journal of Hybrid Information Technology*, *8*, 383–396.

Mirjalili, S., Mirjalili, S. M., & Yang, X.-S. (2014). Binary bat algorithm. *Neural Computing and Applications*, *25*, 663–681.

Misevičius, A. (2015). Using iterated tabu search for the traveling salesman problem. *Information technology and control*, *32*.

Moayedikia, A., Jensen, R., Wiil, U. K., & Forsati, R. (2015). Weighted bee colony algorithm for discrete optimization problems with application to feature selection. *Engineering Applications of Artificial Intelligence*, *44*, 153–167.

Nagata, Y., & Soler, D. (2012). A new genetic algorithm for the asymmetric traveling salesman problem. *Expert Systems with Applications*, *39*, 8947–8953.

Nakamura, R. Y., Pereira, L. A., Costa, K., Rodrigues, D., Papa, J. P., & Yang, X.-S. (2012). Bba: A binary bat algorithm for feature selection. In *Conference on Graphics, Patterns and Images* (pp. 291–297). IEEE.

Nguyen, T.-T., Pan, J.-S., Dao, T.-K., Kuo, M.-Y., Horng, M.-F. et al. (2014). Hybrid bat algorithm with artificial bee colony. In *Intelligent Data analysis and its Applications* (pp. 45–55). Springer.

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

Ouaarab, A., Ahiod, B., & Yang, X.-S. (2014). Discrete cuckoo search algorithm for the travelling salesman problem. *Neural Computing and Applications*, *24*, 1659–1669.

Pan, T.-S., Dao, T.-K., Chu, S.-C. et al. (2015). Hybrid particle swarm optimization with bat algorithm. In *Genetic and Evolutionary Computing* (pp. 37–47). Springer.

Pang, S., Ma, T., & Liu, T. (2015). An improved ant colony optimization with optimal search library for solving the traveling salesman problem. *Journal of Computational and Theoretical Nanoscience*, *12*, 1440–1444.

Parpinelli, R. S., & Lopes, H. S. (2011). New inspirations in swarm intelligence: a survey. *International Journal of Bio-Inspired Computation*, *3*, 1–16.

Pérez, J., Valdez, F., & Castillo, O. (2015). A new bat algorithm with fuzzy logic for dynamical parameter adaptation and its applicability to fuzzy control design. In *Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics* (pp. 65–79). Springer.

Ralphs, T. K., Kopman, L., Pulleyblank, W. R., & Trotter, L. E. (2003). On the capacitated vehicle routing problem. *Mathematical programming*, *94*, 343–359.

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

Rocki, K., & Suda, R. (2012). Accelerating 2-opt and 3-opt local search using gpu in the travelling salesman problem. In *IEEE International Conference on High Performance Computing and Simulation* (pp. 489–495).

Rodriguez, A., Gutierrez, A., Rivera, L., & Ramirez, L. (2015). Rwa: Comparison of genetic algorithms and simulated annealing in dynamic traffic. In *Advanced Computer and Communication Engineering Technology* (pp. 3–14). Springer.

Roeva, O. N., & Fidanova, S. S. (2013). Hybrid bat algorithm for parameter identification of an e. coli cultivation process model. *Biotechnology & Biotechnological Equipment*, *27*, 4323–4326.

Saenphon, T., Phimoltares, S., & Lursinsap, C. (2014). Combining new fast opposite gradient search with ant colony optimization for solving travelling salesman problem. *Engineering Applications of Artificial Intelligence*, *35*, 324–334.

Saji, Y., Riffi, M. E., & Ahiod, B. (2014). Discrete bat-inspired algorithm for travelling salesman problem. In *Second World Conference on Complex Systems* (pp. 28–31). IEEE.

Shi, X. H., Liang, Y. C., Lee, H. P., Lu, C., & Wang, Q. (2007). Particle swarm optimization-based algorithms for tsp and generalized tsp. *Information Processing Letters*, *103*, 169–176.

Singh, A., Thapar, S., Bhatia, A., Singh, S., Goyal, R., & Chen, J. (2015). Disk scheduling using a customized discrete firefly algorithm. *Cogent Engineering*, *2*.

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

Tang, K., Li, Z., Luo, L., & Liu, B. (2015). Multi-strategy adaptive particle swarm optimization for numerical optimization. *Engineering Applications of Artificial Intelligence*, *37*, 9–19.

Tarantilis, C., & 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*, 806–822.

Urrutia, S., Milanés, A., & Løkketangen, A. (2015). A dynamic programming based local search approach for the double traveling salesman problem with multiple stacks. *International Transactions in Operational Research*, *22*, 61–75.

Wang, Y. (2014). The hybrid genetic algorithm with two local optimization strategies for traveling salesman problem. *Computers & Industrial Engineering*, *70*, 124–133.

Wang, Y. (2015). An approximate method to compute a sparse graph for traveling salesman problem. *Expert Systems with Applications*, .

Yang, X.-S. (2009). Firefly algorithms for multimodal optimization. In *Stochastic algorithms: foundations and applications* (pp. 169–178). Springer.

Yang, X.-S. (2010). A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization* (pp. 65–74). Springer.

Yang, X.-S. (2011). Bat algorithm for multi-objective optimisation. *International Journal of Bio-Inspired Computation*, *3*, 267–274.

Yang, X.-S., & He, X. (2013). Bat algorithm: literature review and applications. *International Journal of Bio-Inspired Computation*, *5*, 141–149.

Yang, X.-S., & Hossein Gandomi, A. (2012). Bat algorithm: a novel approach for global engineering optimization. *Engineering Computations*, *29*, 464–483.

Yao, W.-q. (2014). Genetic quantum particle swarm optimization algorithm for solving traveling salesman problems. In *Fuzzy Information & Engineering and Operations Research & Management* (pp. 67–74). Springer.

Yip, P. P., & Pao, Y.-H. (1995). Combinatorial optimization with use of guided evolutionary simulated annealing. *IEEE Transactions on Neural Networks*, *6*, 290–295.

Yousefikhoshbakht, M., & Sedighpour, M. (2013). New imperialist competitive algorithm to solve the travelling salesman problem. *International Journal of Computer Mathematics*, *90*, 1495–1505.

Zhang, J. W., & Wang, G. G. (2012). Image matching using a bat algorithm with mutation. *Applied Mechanics and Materials*, *203*, 88–93.

Zhao, D., Xiong, W., & Shu, Z. (2015). Simulated annealing with a hybrid local search for solving the traveling salesman problem. *Journal of Computational and Theoretical Nanoscience*, *12*, 1165–1169.

Zhou, L., Ding, L., & Qiang, X. (2014). A multi-population discrete firefly algorithm to solve tsp. In *Bio-Inspired Computing-Theories and Applications* (pp. 648–653). Springer.