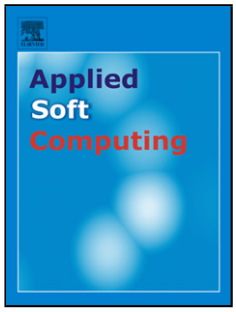


Title: A discrete water cycle algorithm for solving the symmetric and asymmetric traveling salesman problem

Author: Eneko Osaba Javier Del Ser Ali Sadollah Miren Nekane Bilbao David Camacho

PII: S1568-4946(18)30381-8
DOI: https://doi.org/doi:10.1016/j.asoc.2018.06.047
Reference: ASOC 4963

To appear in: *Applied Soft Computing*

Received date: 20-11-2017
Revised date: 11-5-2018
Accepted date: 29-6-2018

Please cite this article as: Eneko Osaba, Javier Del Ser, Ali Sadollah, Miren Nekane Bilbao, David Camacho, A discrete water cycle algorithm for solving the symmetric and asymmetric traveling salesman problem, *<![CDATA[Applied Soft Computing Journal]]>* (2018), https://doi.org/10.1016/j.asoc.2018.06.047

A Discrete Water Cycle Algorithm for Solving the Symmetric and Asymmetric Traveling Salesman Problem

Eneko Osaba[a,*], Javier Del Ser[a,b,c], Ali Sadollah[d],
Miren Nekane Bilbao[b], and David Camacho[e]

[a] *TECNALIA Research and Innovation, 48160 Derio, Spain*
[b] *University of the Basque Country (UPV/EHU), 48013 Bilbao, Spain*
[c] *Basque Center for Applied Mathematics (BCAM), 48009 Bilbao, Spain*
[d] *Department of Mechanical Engineering, Sharif University of Technology, 11365-11155 Tehran, Iran*
[e] *Escuela Politecnica Superior, Universidad Autonoma de Madrid, 28049 Madrid, Spain*

Abstract

The Water Cycle Algorithm (WCA) is a nature-inspired meta-heuristic recently contributed to the community in 2012, which finds its motivation in the natural surface runoff phase in water cycle process and on how streams and rivers flow into the sea. This method has been so far successfully applied to many engineering applications, spread over a wide variety of application fields. In this paper an enhanced discrete version of the WCA (coined as DWCA) is proposed for solving the Symmetric and Asymmetric Traveling Salesman Problem. Aimed at proving that the developed approach is a promising approximation method for solving this family of optimization problems, the designed solver has been tested over 33 problem datasets, comparing the obtained outcomes with the ones got by six different algorithmic counterparts from the related literature: Genetic Algorithm, Island-based Genetic Algorithm, Evolutionary Simulated Annealing, Bat Algorithm, Firefly Algorithm and Imperialist Competitive Algorithm. Furthermore, the statistical significance of the performance gaps found in this benchmark is validated based on the results from non-parametric tests, not only in terms of optimality but also in regards to convergence speed. We conclude that the proposed DWCA approach outperforms – with statistical significance – any other optimization technique in the benchmark in terms of both computation metrics.

Keywords: Routing problems, Water Cycle Algorithm, Traveling Salesman Problem, combinatorial optimization

*Corresponding author: Dr. Eneko Osaba. TECNALIA RESEARCH & INNOVATION, P. Tecnologico Bizkaia, Ed. 700. 48170 Derio, Bizkaia, Spain. Phone: +34 946 430 850

Email addresses: eneko.osaba@tecnalia.com (Eneko Osaba), javier.delser@tecnalia.com (Javier Del Ser), ali.sadollah@yahoo.com (Ali Sadollah), nekane.bilbao@ehu.es (Miren Nekane Bilbao), david.camacho@uam.com (and David Camacho)

1. Introduction

Routing problems are among the most studied paradigms in operations research and optimization. The reason for their importance and attractiveness is twofold: their complexity, which makes such problems difficult to solve even for simplistic optimization metrics and medium-sized datasets; and its inherently practical nature and social interest, which spans the applicability of routing algorithms to scenarios not only related to business and logistics, but also of utility in leisure and tourism, among others. Indeed, the formulation of routing problems often draws inspiration from real-world logistic situations, hence solving a routing problem usually implies a straightforward business and/or social benefit. However, solving these problems efficiently remains a challenge for the scientific community due to their NP-hard nature, which motivates the adoption of very diverse technical approaches to address this optimization task in a computationally affordable fashion. The problem gets even more involved when bearing in mind the rich literature in regards to different formulations of routing problems, among which the so-called Traveling Salesman Problem (TSP, [48]) and the Vehicle Routing Problem (VRP, [14]) can be considered, arguably, as the most studied ones.

In this context, many different optimization methods have been hitherto contributed by the research community to deal with routing problems. The most studied and used schemes are exact methods [45, 46], heuristics, and meta-heuristics. This paper is focused in the latter category, whose reported optimization techniques have demonstrated a notable efficiency in solving efficiently routing problems, particularly in the last decade. Some examples of such techniques are Tabu Search (TS, [27]) and Simulated Annealing (SA, [42]) as local search-based methods, and Genetic Algorithm (GA, [28, 18]), Ant Colony Optimization (ACO, [8, 83]) or Particle Swarm Optimization (PSO, [41, 73]) as the most well-known population-based global optimizers. Despite the existence of these classical methods, the modification and application of new solvers to routing problems is on the spotlight of the scientific community. As a consequence of this ever-growing general interest in optimization methods, many successful meta-heuristic algorithms have been proposed in the last decade, such as the league championship algorithm [40], the Imperialist Competitive Algorithm (ICA, [6]), Cuckoo Search (CS, [79]), or the Firefly Algorithm (FA, [77, 84]), among others.

In line with this research trend, this paper focuses on a recently proposed nature-inspired meta-heuristic algorithm, coined as the Water Cycle Algorithm (WCA, [20]). The search process of WCA finds its motivation in the surface runoff phase in water cycle process in nature, and by the physical mechanisms by which streams and rivers flow into the sea. As such, a population of streams forms swarms in WCA flowing into the rivers and sea. Movements from streams to rivers and from rivers to the sea, or even the fact that some streams may directly flow into the sea, provide an indirect drift towards the best solution. The WCA attempts at modeling the entire water cycle process, yet in practice its standard version mostly simulates the surface runoff phenomenon. Therefore, considering other phases in water cycle, there would be room for further modifications in this matter.

We next provide an outlook on the state of the art around WCA, TSP and applications so as to best contextualize the novel ingredients of the present study.

1.1. Literature Review and Motivation on WCA and TSP

There are several applications of WCA in the literature for engineering purposes. To mention a few, Haddad et al. [33] utilized the WCA for finding the optimal operation regime of reservoir systems. Their obtained optimization results demonstrated the high efficiency and reliability of the WCA for tackling reservoir operation problems. Ghaffarzadeh [26] utilized the WCA as a stabilizer for the optimal design of power systems. Baghipour et al. [7] used the WCA for finding an optimal number, location, and size of multiple types of distributed generation units in distribution system. Kong et al. [44] proposed an enhanced WCA (EWCA) for solving multi-reservoir systems with complex constraints.

More recently, the WCA has been applied to urban traffic management for finding optimal scheduling of traffic light considering traffic congestions in traffic networks along with other optimizers [25]. The WCA showed lower traffic congestion especially for larger traffic networks compared with the reported optimization methods. In energy, the WCA was utilized for extracting the optimal parameter setting of triple-junction solar cells (TJSC) operating under various irradiance and temperature levels [67]. Newly, a Gaussian bare-bones WCA (GBWCA) has been developed by Heidari and his colleagues [34]. The proposed algorithm has been applied to optimal reactive power dispatch problem. Resistive losses and voltage deviations were the objectives in reactive power dispatch problem. The proposed WCA outperformed some state-of-the-art optimization methods.

Besides application-oriented contributions, many variants of the standard WCA have been extensively proposed since this algorithm was first proposed. For instance, Guney and Basbug [31] proposed quantized WCA (QWCA), and applied this modified solver for the synthesis of antenna array patterns. The internal quantization mechanism of QWCA was utilized to achieve digital values matching to the discrete values of the phase shifter instead of the simple rounding up/down routines after optimization. Praepanichawat et al. [65] proposed hybrid version of WCA by combining the WCA with artificial bee colony for optimally solving allocation problems. Recently, Guo and Li [32] utilized and combined the ER-WCA for parameter estimation of Newtons rings. The experimental results evinced that the efficiency of their proposed approach was much higher than that of fractional Fourier transform based methods. Pahnehkolaei et al. [63] derived and included a local optimization operator coined as gradient-based approach in the standard WCA. The idea exploited the concept of moving individuals along the steepest direction slope under a certain criterion. The proposed gradient-based WCA (GWCA) was employed to obtain the optimal gains of a backstepping controller for chaos suppression problems.

On another note, the TSP has been central in many optimization benchmarks ever since the mid past century [16], and has as such been resorted frequently to empirically validate the performance of new discrete optimization algorithms and combinatorial optimizers. Indeed, various state-of-the-art optimizers have hitherto been applied to the TSP such as the aforementioned GA [29, 59], TS [21, 43] or SA [53, 1], withstanding to date as the optimization alternatives of choice in practical TSP scenarios [85, 49, 75]. ACO and PSO have also been widely applied to the TSP [30, 55, 54]. The TSP has also been used as a globally accepted optimization benchmark for recently proposed nature-inspired approaches such as the BA [61], FA [37], CS [62], and ICA [5]. Furthermore,

it is interesting to mention hybrid approaches developed and tested with the TSP. The work presented in [68] is an example of a hybrid method encompassing ACO and gradient search. Likewise, in [52] a hybrid method based on PSO, ACO and the 3-Opt algorithm was presented. Other examples blend together SA and local search operators [86], or concepts drawn from GA, SA, PSO and ACO into a single TSP solver [13]. The comprehensive surveys presented in [4] and [35] are highly recommended material for readers interested in further information about the TSP and related solvers.

An interesting point, which must be regarded as one of the key contributions of this work, is the discretization of the WCA so as to undertake combinatorial optimization problems. The naïve version of WCA was originally conceived to address continuous-variable problems. Subsequently, WCA has been adapted to discrete problems in some few works. Intuitively, the discretization strategy is a crucial design step for the final performance of the designed optimization algorithm. In [24], for example, a discrete version of WCA is presented for solving a manufacturing rescheduling problem allowing for new job insertion under two conflicting objectives: 1) the total flow of the system, and 2) its stability. In this work discretization was accomplished by just performing a exchange of solution parts between streams, rivers and sea triggered by a uniformly distributed binary variable. Recently, in [76] a binary encoded WCA is proposed for dealing with the construction of Bayesian network structures. Despite the nature of the problem tackled, this work also endows the WCA with some novel discrete operators for computing the position of candidate solutions. All in all, research on discrete versions of the WCA have not grown at the same pace than its continuous variable counterpart.

In line with the last cited work on binary-encoded WCA, transfer functions are often used in the literature for the discretization of continuous optimization problems. Being v-shape and s-shape two of the most widely known alternatives [57], these functions permit to convert continuous optimization methods to binary approaches, respecting the original search algorithm of the technique at hand. These functions have been used frequently in the literature for this purpose, such as [69] in which the performance of eight different functions is tested using the Gravitational Search Algorithm and PSO as example metaheuristics. Another interesting application was reported in [58], in which a v-shaped function is employed for implementing a Binary Bat Algorithm. The same function is used for the binary variant of the Dragonfly Algorithm proposed in [56]. Likewise, in [70] the non-unicost set covering problem is solved by using a Cuckoo Search method and a Black Hole optimization technique. For these approaches transfers functions are also utilized for efficiently handling the binary nature of the problem. The same problem is also tackled using a Firefly Algorithm with transfer function in [15].

Alternatively to transfer functions, in this work we embrace the well-known path representation [47] for encoding solutions to both TSP and ATSP. Each solution is encoded as a permutation of numbers, which represents the order in which nodes are visited. This encoding approach is the most used in the literature for solving TSP and ATSP, and its natural representation of paths allows for the development of search operators well-suited to such problems (e.g. permutation, swapping) as the ones that described in forthcoming sections.

As can be inferred from the literature reviewed above, WCA has been applied to

a wide variety of optimization problems, in which it has shown to perform better than some of the most renowned meta-heuristic solvers. Based on these empirical findings, WCA has emerged as a nature-inspired method that efficiently balances exploration and exploitation, encompassing both cooperative mechanisms and local search operations at an affordable computational effort. Although WCA has been used in problems stemming from different disciplines (as has been evinced above), this heuristic algorithm has not been applied to routing problems to date. All its computational advantages, along with the lack of research and the flourishing interest in this technique shown by the community are the main reasons for the work described in this manuscript. Additionally, WCA is easy to implement and is controlled by a reasonable number of parameters. These properties ease the derivation and inclusion of the herein proposed novel *inclination* mechanism into the nave WCA search procedure, which improves further its search capability.

1.2. Contribution

Bearing the above state of the art in mind, there is no precedent study on the adaptation of WCA for TSP problems, nor has this advance technique been profusely explored for discrete optimization problems. This paper covers this lack of research by elaborating on a novel discrete version of the WCA (denoted hereafter as DWCA) for solving both classical (TSP) and asymmetric (ATSP) formulations of this routing problem. The main goal of the research work presented in this manuscript is to gauge whether the proposed DWCA is a promising approach for solving both the TSP and the ATSP. To this end, results obtained by the proposed DWCA will be compared to those obtained by three different classical techniques: GA, evolutionary SA (ESA) [81], and an Island based Genetic Algorithm (IGA) [2]. Furthermore, the performance of the DWCA is also compared with the discrete versions of three modern bio-inspired approaches: FA, BA and ICA. All these three techniques have demonstrated a notable performance in recent years, grasping the interest of the scientific community. This comparison builds upon and extends significantly the findings reported recently in [61]. Experiments have been conducted over 33 TSP-ATSP datasets extracted from the related literature and available publicly. With the aim of obtaining fair and objective conclusions, the statistical significance of the observed performance gaps will be analyzed by means of two different statistical tests along with the conventional methodology based on descriptive performance statistics. As the obtained results clearly show, the proposed DWCA dominates over the rest of heuristics included in the benchmark for the majority of simulated TSP/ATSP datasets, not only in terms of the optimality of the produced routes, but also in what refers to its computational efficiency under the same termination criterion.

The remaining of this manuscript is organized as follows. For the sake of completeness Section 2 formulates mathematically both TSP and ATSP problems, whereas Sections 3 and 4 describe the standard WCA and the proposed discrete version of the WCA, respectively. Numerical results obtained by reported optimizers and the proposed DWCA are tabulated and discussed in Sections 5 and 6. Finally, conclusions and further works are outlined in Section 7.

2. Problem Statement

The TSP and ATSP can be represented as a complete graph $\mathcal{G} \doteq (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} \doteq \{v_1, v_2, \ldots, v_N\}$ denotes the group of vertex that represent the nodes of the graph, and $\mathcal{A} \doteq \{(v_i, v_j) : v_i, v_j \in \mathcal{V} \times \mathcal{V},\ i, j \in \{1, \ldots, N\} \times \{1, \ldots, N\}, i \neq j\}$ is the set of edges connecting every pair of nodes in $\mathcal{V}$. Furthermore, each edge has an associated weight $d_{ij} \in \mathbb{R}^+$, standing for the traveling cost thorough this arc. In the symmetric version of the problem $d_{ij} = d_{ji}$, meaning that the cost of going from one node to another is the same as its reverse trip. Despite the eventual existence of some edges in which $d_{ij} = d_{ji}$, ATSP refers to the case where $d_{ij} \neq d_{ji}$ for at least one edge in $\mathcal{A}$.

The optimization problem characterizing both the TSP and ATSP hinges on the discovery of a route that visits every node once and only once, that is, a Hamiltonian cycle in the graph G, whose total cost (aggregated over all vertex in $\mathcal{V}$ that compose the route) is minimum. Posed formally, this generic optimization problem can be formulated as

$$\underset{\mathbf{X}}{\text{minimize}} \quad f(\mathbf{X}) = \sum_{i=1}^{N} \sum_{\substack{j=1 \\ i \neq j}}^{N} d_{ij} x_{ij} \tag{1a}$$

$$\text{subject to} \quad \sum\nolimits_{\substack{j=1 \\ i \neq j}}^{N} x_{ij} = 1, \quad \forall j \in \{1, \ldots, N\}, \tag{1b}$$

$$\sum\nolimits_{\substack{i=1 \\ i \neq j}}^{N} x_{ij} = 1, \quad \forall i \in \{1, \ldots, N\}, \tag{1c}$$

$$\sum\nolimits_{\substack{i \in \mathcal{S} \\ j \in \mathcal{S} \\ i \neq j}} x_{ij} \geq 1, \quad \forall \mathcal{S} \subset \mathcal{V}, \tag{1d}$$

where $\mathbf{X} \doteq [x_{ij}]$ is a $N \times N$ binary matrix whose entry $x_{ij} \in \{0, 1\}$ takes value 1 if edge (i, j) is used in the solution. With all this notation in mind, the objective function represented in Expression (1a) is the sum of weights associated to all the arcs contained in the solution, i.e., the total cost of the route represented by $\mathbf{X}$. Expressions (1b) and (1c) indicate that each node has to be visited only once. Finally, (1d) ensures the absence of sub-tours, and forces that any subset of nodes S has to be abandoned at least one time. This restriction is essential to avoid cycles along the route.

3. Water Cycle Algorithm

The WCA solver mimics the phenomena of flowing rivers and streams toward the sea and derived by the observation of water cycle process in nature. Assumed that raining or precipitation occurs, a P-sized initial population of design variables (i.e. a population of streams) is generated at random. The best individual (i.e. the best stream) in terms of minimal cost (for minimization problem) is selected to represent the *sea* [20]. Then, a number of good streams (that is, those whose cost function values are close to the current best record) are elected as *rivers*, while the rest of streams do not change and just flow into the rivers and sea. Hence, in a N-dimensional optimization

problem, a stream is a $1 \times N$ array defined as

$$\mathbf{x}^p = [x_1^p, x_2^p, x_3^p, \ldots, x_N^p] \text{ (candidate)}, \tag{2}$$

where $p \in \{1, \ldots, P\}$, and N is the number of design variables (problem dimension). As anticipated above, an initial population of streams represented by a $P \times N$ matrix is first generated. Then, a P_{sr}-sized subset $\mathcal{P}_{sr} \subseteq \mathcal{P} \doteq \{1, \ldots, P\}$ from the best individuals are selected to be the sea (one single sea given by the stream featuring the best fitness value) and rivers (the rest of streams in the aforementioned subset). The remaining population $\mathcal{P}_{str} \doteq \mathcal{P} - \mathcal{P}_{sr}$ represents streams that may flow into rivers or directly to the sea, with cardinality $P_{str} \doteq P - P_{sr}$.

Judging by the value of flow intensity, each river absorbs water from the streams. The amount of water entering a river and/or the sea, hence, varies from stream to stream. In addition, rivers flow to the sea which is the most downhill location. The designated streams flowing into each river and sea are given by

$$NS_p = \left\lfloor \frac{C_p}{\sum_{p \in N_{sr}} C_p} \cdot N_{str} \right\rfloor, \quad p \in \mathcal{P}_{sr} \tag{3}$$

where $C_p = f(\mathbf{x}^p) - \min_{p' \in \mathcal{P}_{str}} f(\mathbf{x}^{p'}) \; \forall p \in \mathcal{P}_{sr}$, and NS_n is the number of streams which flow to the specific rivers and sea. As can be seen in nature, streams are created from the water of raining and join each other to generate new rivers. Some streams may even flow directly to the sea. All rivers and streams end up in the sea that corresponds to the current best solution. Let us assume that there are P streams of which $P_{sr} - 1$ are selected as rivers, and one is selected as the sea (e.g. the candidate in the p_{sea}-th position within the population). For the exploitation phase in the WCA search process, new positions for streams $p_{str} \in \mathcal{P}_{str}$ and rivers $p_r \in \mathcal{P}_{sr} - \{p_s\}$ evolve as:

$$\mathbf{x}^{p_{str}}(t+1) = \mathbf{x}^{p_{str}}(t) + \text{Uniform}(0, C) \cdot \left(\mathbf{x}^{\lambda(p_{str})}(t) - \mathbf{x}^{p_{str}}(t) \right), \tag{4}$$

$$\mathbf{x}^{p_r}(t+1) = \mathbf{x}^{p_r}(t) + \text{Uniform}(0, C) \cdot \left(\mathbf{x}^{p_{sea}}(t) - \mathbf{x}^{p_r}(t) \right), \tag{5}$$

where t stands for the iteration index (i.e. generation), $\lambda : \mathcal{P}_{str} \mapsto \mathcal{P}_{sr}$ is a mapping function from streams to rivers and sea, and $\text{Uniform}(0, C)$ denotes a realization of an uniformly distributed random variable between zero and the real-valued parameter $C \in [1, 2]$. If the solution given by a stream is better than its connecting river, the positions of river and stream are exchanged (i.e., the stream becomes a river and the river becomes a stream). A similar exchange should be applied for a river and the sea. Figure 1 represents a possible exchange of a stream which becomes the best solution among other streams and its river.

The evaporation process operator also is introduced to avoid premature convergence to local optima during the exploitation phase of the search process. Basically, evaporation causes sea water to evaporate as rivers/streams flow to the sea. This leads to new incoming precipitations. Therefore, the algorithm should check whether the river/stream is close enough to the sea to make the evaporation process occur. In particular the seminal definition of the WCA solver proposes to implement evaporation and raining whenever the Euclidean distance between the sea and the river/stream at

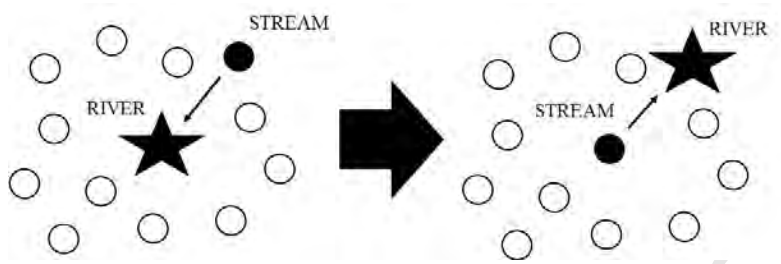


Figure 1: Exchange of the position of the river and the stream. The *star* symbol represents the river, whereas black circles stand for the best stream.

hand is less than a predefined distance d_{max} or, alternatively, at a minimum fixed rate $R_{ev} \in [0, 1]$. Evaporation consists of an adaptive decrease of the value of d_{max} along iterations. A large value for d_{max} prevents extra searches and small values encourage the search intensity near the sea. Therefore, d_{max} controls the search intensity near the sea (i.e., best obtained solution); indeed, the evaporation operator is responsible for the search exploration. Without loss of generality in regards to other progression laws, this decrease of d_{max} can be done as

$$d_{max}(t+1) = d_{max}(t) - (d_{max}(t)/T)\,, \tag{6}$$

where $t \in \{1, \ldots, T\}$ and T denotes the maximum number of iterations. After the evaporation condition, the raining process is applied and new streams are formed in different locations similarly to the well-known mutation operator in GA. Hence, in the newly generated sub-population the best stream will play a role as a new river and other streams move toward their new river. This condition will also hold for streams that directly flow into the sea. Locations of the newly formed streams are computed as

$$x_n^p = \text{Uniform}(LB_n, UB_n), \tag{7}$$

where x_n^p denotes the n-th variable of the stream p to be formed, and (LB_n, UB_n) are the lower and upper bounds defined for x_n^p in the statement of the given problem, respectively. The newly formed stream is considered as a river flowing into the sea. The rest of new streams are assumed to either flow into rivers or directly into the sea. For further clarifying the overall process, Algorithm 1 describes the steps of WCA in a formal fashion, where it is assumed that the *sea* is replaced with the newly formed river upon evaporation and raining. Finally, the schematic view of the base WCA is illustrated on Figure 2

4. Proposed Discrete Water Cycle Algorithm (DWCA) for the TSP/ATSP

As can be inferred from the above explanation, the original WCA solver was mainly intended for continuous optimization problems. Therefore, this heuristic technique must be adapted in order to address combinatorial optimization problems such as the TSP and ATSP tackled in this paper.

Algorithm 1: Standard WCA algorithm

Data: Parameters P (population size), P_{sr} (number of rivers and sea), d_{max} (maximum distance for evaporation), R_{ev} (minimum rate for evaporation), C (amplitude for the position update), T (maximum number of generations), $f(\mathbf{x})$ (fitness function), $\{(LB_n, UB_n)\}_{n=1}^{N}$ (lower and upper bounds for the optimization variables)

Result: Optimal solution $\mathbf{x}^*$, minimum fitness value $f(\mathbf{x}^*)$

Randomly initialize the initial population considering $\{(LB_n, UB_n)\}_{n=1}^{N}$, and select streams (subset $\mathcal{P}_{str}$), rivers (corr. $\mathcal{P}_r$) and sea taking into account the value of P_{sr}
Calculate the cost value C_p of each stream $p \in \mathcal{P}_{str}$, and determine the designated streams that flow into rivers and sea using Expression (3)
Set $t = 1$ (iteration number)
while $t \leq T$ or termination criterion met **do**
    Update the location of streams with respect to their river/sea as per Expression (4)
    Update the location of streams with respect to the sea as per Expression (5)
    If the stream/river has a better fitness metric than its counterpart, exchange roles
    for $p \in \mathcal{P}_r \cup \mathcal{P}_{str}$ **do**
        if Distance$(\mathbf{x}^p(t), \mathbf{x}^{p_{sea}}(t)) < d_{max}(t)$ **or** Uniform$(0,1) < R_{ev}$ **then**
            Decrease the value of d_{max} as per Expression (6)
            Replace the *sea* with a new stream/river with location given by (7)
            break
        end
    end
    Recompute costs, subsets $\mathcal{P}_{str}$ and $\mathcal{P}_{sr}$, and the best individual (*sea*) using the updated streams/rivers
    Set $t = t + 1$
end
The best solution and its fitness are given by the p_{sea}-th candidate (namely, the sea).

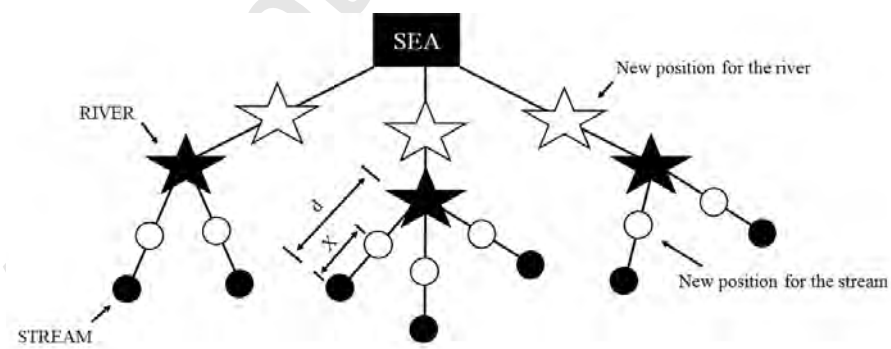


Figure 2: Schematic diagram illustrating the baseline WCA search procedure.

When dealing with these particular problem datasets, solutions representing routes can be encoded by following very diverse strategies. As has been mentioned before, in this work the frequently utilized path encoding has been selected: each solution is

encoded as a permutation of numbers representing the order in which the nodes $\mathcal{V}$ are visited. For example, in a given 12-node dataset of the TSP a feasible solution can be encoded as $\mathbf{x} = [11, 2, 7, 6, 9, 0, 10, 8, 1, 3, 5, 4]$, i.e. node 11 is visited first, followed by nodes 2, 7 and so forth. Each individual of the population in the WCA embraces this encoding approach to represent a complete and feasible solution for the TSP and ATSP. Furthermore, the objective function used, which is also important for calculating the nature of each stream, is the total cost of a complete route give in Expression (1a).

Arguably, the most difficult aspect when adapting the WCA to the considered problems is to design heuristic operators resembling how streams flow into rivers and sea, while ensuring that they contribute effectively to the search problem under study. To this end, in the proposed DWCA the measure of distance between the streams and their corresponding rivers or sea is selected to be the well-known Hamming distance between the two compared individuals (routes). The Hamming distance is given by the number of non-corresponding elements in the sequence of both individuals, e.g. if the following vectors represent two feasible routes:

$$\mathbf{x}^p = [9, 7, 3, 4, 1, 5, 0, 2, 6, 8],$$
$$\mathbf{x}^{p'} = [8, 7, 3, 4, 1, 5, 0, 9, 6, 2],$$

their Hamming Distance $D_H(\mathbf{x}^p, \mathbf{x}^{p'})$ would be equal to 3. Once the distance between the two individuals has been computed, the movement of the stream or river must be performed. In DWCA two well-known movement operators have been used depending on the distance between the river/stream and the sea, or the stream and its corresponding river:

- *Insertion:* this is one of the most frequently used functions for solving combinatorial optimization problems of different nature. Specifically, it selects and extracts one randomly chosen node from the route. Afterwards, this node is re-inserted again in the route in a randomly selected position.

- *2-opt:* This operator first proposed in [51] has been extensively applied in different kinds of routing problems such as the TSP and the VRP [74, 9]. The main design principle behind this operator is to randomly eliminate two arcs within the existing route, in order to create two new arcs, avoiding the generation of sub-tours.

Therefore, following the aforementioned behavior of Hamming Distance and the rules of the standard WCA, the movement performed by each stream $p_{str} \in \mathcal{P}_{str}$ to its corresponding river or sea $\lambda(p_{str}$ at each generation $t \in \{1, \ldots, T\}$ can be defined as:

$$\mathbf{x}^{p_{str}}(t+1) = \Psi\left(\mathbf{x}^{p_{str}}(t), \min\left\{N, \left\lfloor \text{Uniform}(0, C) \cdot D_H(\mathbf{x}^{p_{str}}(t), \mathbf{x}^{\lambda(p_{str})}(t))\right\rfloor\right\}\right), \quad (8)$$

where $\Psi(\mathbf{x}, M)$ denotes the movement function that can be either the above explained *insertion* or *2-opt* operators, parametrized with the number of times M this operator is applied to route $\mathbf{x}$. The best route among all M movements performed over $\mathbf{x}$ is selected as the output of the operator. The same rationale holds for the movement of a stream or a river directly flowing into the sea, which is set identically as the one explained above by replacing $\mathbf{x}^{\lambda(p_{str})}(t)$ with $\mathbf{x}^{p_{sea}}(t)$.

In addition, one of the main improvements of the developed discrete version of the WCA respect to its standard version is its ability to implement different movement policies depending on the distance between the individuals. Specifically, every time a river or stream is ready to perform a movement, its so-called *inclination* $\xi(\mathbf{x}, \mathbf{x}')$ is computed as the ratio between the distance from the stream/river $\mathbf{x}$ to its designated river/sea $\mathbf{x}'$ and its maximum given by N (i.e. the size of the problem). For a stream/river $\mathbf{x}$ leading to river/sea $\mathbf{x}'$, this yields

$$\xi(\mathbf{x}, \mathbf{x}') = \frac{D_H(\mathbf{x}^{p_{str}}(t), \mathbf{x}^{\lambda(p_{str})}(t))}{N}, \tag{9}$$

By assuming that the bigger the distance $D_H(\cdot,\cdot)$ is, the higher the inclination $\xi(\cdot,\cdot)$ should be, a *fast move* should be enforced with a higher probability if the inclination is high. Conversely, if the distance is small the inclination decreases, which suggests that the stream is potentially in a promising area of the solution space. Thus, this individual should performs a *slow move* with higher probability. Due to the nature of the considered movement functions, *insertion* can be regarded as a *slow move*, whereas *2-opt* can be deemed a *fast move*. Therefore, *2-opt* will be adopted as the selected movement function with probability $\xi(\cdot,\cdot)$, and *insertion* with probability $1 - \xi(\cdot,\cdot)$. This simple but effective improvement permits the streams to crawl the space of solutions in different ways along the execution, exploring different neighborhood structures. The main idea behind this improvement is to enhance the exploration ability of the DWCA method, with the ultimate aim at reaching routes of higher quality. Figure 3 illustrates this innovative concept schematically.

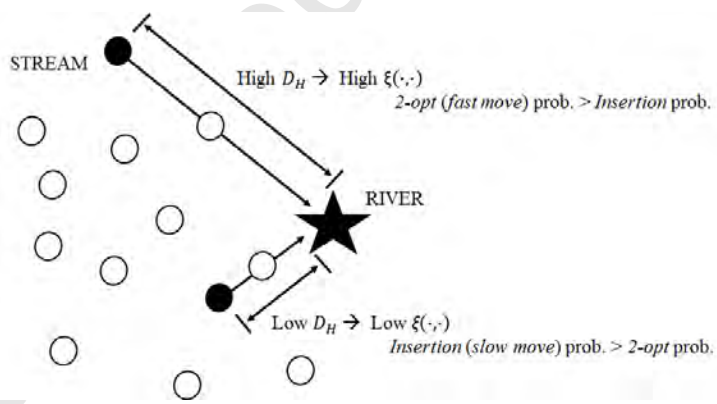


Figure 3: Schematic example of the proposed inclination mechanism.

The last aspect to detail is the evaporation condition and raining procedures. In this sense, the philosophy of both processes remains in the same way as the basic concept used in the WCA. Once the evaporation condition is satisfied, the raining process is executed. As in the basic version of the WCA, this procedure acts similarly to mutation operator in GA. More specifically, the raining process of the developed DWCA is a number R of consecutive random *insertion* movements. The overall algorithmic flow

of the proposed DWCA approach is schematically shown in Figure 4 and described in Algorithm 2.

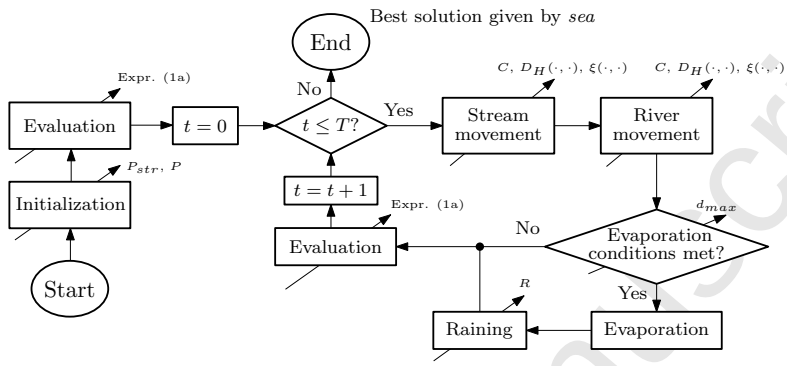


Figure 4: Block diagram representing the overall processing flow of the proposed DWCA solver.

Algorithm 2: Proposed DWCA algorithm

Data: Graph $\mathcal{G}$ representing the TSP/ATSP instance, parameters P (population size), P_{sr} (number of rivers and sea), d_{max} (maximum distance for evaporation), R_{ev} (minimum rate for evaporation), C (amplitude for the position update), T (maximum number of generations), R (number of repetitions for raining phase)

Result: Optimal solution $\mathbf{x}^*$, minimum fitness value $f(\mathbf{x}^*)$

Randomly initialize the initial population with permutations of the set $\{1, \ldots, N\}$, and select streams (subset $\mathcal{P}_{str}$), rivers (corr. $\mathcal{P}_r$) and sea taking into account P_{sr}
Calculate the cost value C_p of each stream $p \in \mathcal{P}_{str}$ with $f(\mathbf{x})$ (fitness of a route) as per (1a), and designate streams that flow into rivers and sea using Expression (3)
Set $t = 1$ (iteration number)
while $t \leq T$ or termination criterion met **do**
    Update the location of streams $\mathbf{x}$ with respect to their river/sea $\mathbf{x}'$ as per Expression (8) using *2-opt* as $\Psi(\cdot,\cdot)$ (with probability $\xi(\cdot,\cdot)$) or *insertion* otherwise
    If the stream/river has a better fitness metric than its counterpart, exchange roles
    for $p \in \mathcal{P}_r \cup \mathcal{P}_{str}$ **do**
        if $D_H(\mathbf{x}^p(t), \mathbf{x}^{p_{sea}}(t)) < d_{max}(t)$ **or** $\text{Uniform}(0,1) < R_{ev}$ **then**
            Decrease the value of d_{max} as per Expression (6)
            Replace $\mathbf{x}^{p_{sea}}(t)$ with a new stream/river with location given by incrementally performing R insertion movements onto $\mathbf{x}^{p_{sea}}(t)$
            break
        end
    end
    Recompute $\mathcal{P}_{str}$ and $\mathcal{P}_{sr}$ using the updated streams/rivers
end
The best solution and its fitness are given by the best in $\{\mathbf{x}^{p_{sea}}(t)\}_{t=1}^{T}$ (the *best* sea).

5. Experimental Setup

The performance of the proposed DWCA approach has been gauged over 33 well-known and contrasted TSP/ATSP datasets. More concretely, 19 datasets have been used for the TSP and 14 benchmarks have been investigated for the ATSP, all extracted from the renowned TSPLIB repository [66] often used in routing based studies. The size of the considered datasets is between 17 and 264 nodes. Results obtained by the developed technique over these problem datasets are compared to those achieved by three classical search techniques from the literature – namely, GA, ESA and IGA –, as well as to recently proposed nature-inspired algorithms based on the BA, FA, and ICA heuristics. Before proceeding with further configuration details of the performed experiments, the basic principles of each approach used in the benchmark are next summarized:

- *Evolutionary Simulated Annealing* (ESA): SA is one of the most used local search method in the literature, and is based on the physical principles explaining the metal cooling process. For the sake of fairness, an evolutionary version of the SA is used (ESA [81]) so as to consider a population-based method roughly similar to the rest of considered solvers.

- *Genetic Algorithm* (GA): this solver emulates the genetic process of living organisms and the laws ruling the evolution of species to yield one of the most utilized heuristic methods for solving optimization problems [36]. In order to overcome the drawbacks of the standard GA, a parallel GA (PGA) was first proposed in [64], which provides substantial gains in performance. Three different parallel GA categories can be found in the literature [3], namely, Panmitic model, Fine Grain, and Island Model. The last category consisting on multiple populations which evolve independently, exchanging individuals at certain moments during their evolution. This is the parallel GA adopted in this benchmark, which will be hereafter referred to as IGA.

- *Firefly Algorithm*: FA was proposed in [77] hinging on the flashing behavior of fireflies, which acts as a signal system to attract other fireflies. This meta-heuristic optimization algorithm has been also applied to a wide range of optimization fields and problems since its proposal, with extensive surveys [23, 22] evincing the momentum gained by this solver within the community.

- *Bat Algorithm*: BA [78] is based on the echolocation capability of microbats, which are able to find their prey and discriminate different kinds of insects even in complete darkness. Similarly to FA, BA is also a successful meta-heuristic, which has scored a promising performance in a variety of optimization problems [80, 12].

- *Imperialist Competitive Algorithm*: this optimizer finds its roots in the concept of imperialism, dividing the whole population in independent empires [6]. Individuals of the empire are divided into two categories depending on their roles inside the population: imperialist states (i.e. the best country within the empire) and colonies. Thus, colonies make their movements on the space of solutions based on the imperialist state of their corresponding empire. At the same time, empires fight to

each other aimed at conquering the weakest colonies of the rest of empires. Several application scenarios have so far resorted to ICA, such as [39, 38].

At this point it is important to mention that results obtained by all the comparison techniques (GA, IGA, ESA, FA, BA, and ICA) have been drawn from the recently published work in [61]. For this reason, and because of the different computers used for running each technique, the computational effort is evaluated by inspecting the convergence rate of each technique (namely, the number of function evaluations needed for converging under a given stopping criterion for the search process), instead of measuring the runtime. As proven in several studies, this is a fair, implementation-agnostic methodology to compare techniques that have been run in different computing platforms [72, 10]. Furthermore, 20 independent runs have been executed for each (*problem*, *technique*) combination so as to provide statistically reliable insights on the performance of every technique over each problem instance. Furthermore, two different statistical tests are utilized to verify whether the performance gaps discovered among the considered techniques are statistically relevant.

Based on the good practices described in [60], similar parameters and functions have been used in all heuristics included in the benchmark. The goal is to determine which technique produces better results using the same stopping condition: each run finishes when there are $N + \sum_{k=1}^{N}$ generations without improvements in the best solution found, where N is the size of the problem [61]. This termination criterion is the same used for the rest of the methods used in the experimentation. Table 1 summarizes the parameters configured for GA, IGA and ESA. It can be observed that ESA counts with two different successor functions, meaning that every individual has its own randomly assigned successor function. A similar procedure has been considered in GA with the crossover function, and in PGA and IGA with the mutation function. Finally, the guidelines given in [50, 82] have been followed for the parameter setting of FA, BA and ICA. Furthermore, the same population size (namely, $P = 48$ individuals) has been used also for DWCA, FA, BA and ICA. The Hamming distance function $D_H(\cdot, \cdot)$ has been used for the movement phase in all considered solvers.

GA		IGA		ESA	
Parameter	Value	Parameter	Value	Parameter	Value
Population size	50	Population size	4 subpob. of 13 individuals	Population size	50
Crossover function	OX	Crossover functions	OX & OBX	Successor functions	2-opt & Insertion
Mutation functions	Insertion & 3-opt	Mutation functions	Insertion & 3-opt	Temperature	$-sup\Delta f/ln(p)$
Cross. prob.	0.95	Cross. prob.	[0.95, 0.9, 0.8, 0.75]	Cooling constant	0.95
Mut. 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 [11]		

Table 1: Parameter setting of GA, IGA and ESA for the TSP and ATSP. OX: Order Crossover ([17]). OBX: Order Based Crossover ([71]). $-\text{sup}\Delta f$ is the difference in the objective function of the best and the worse individuals of the initial population, and $p = 0.95$.

Regarding DWCA, the number of rivers and sea P_{sr} has been established to 10 individuals (approximately 20% of the entire population), leading to a number of streams equal to $P_{str} = 38$. Values of d_{max} and R have been set to 5% and Uniform$(0, \lfloor 0.5N \rfloor)$ with respect to the problem size N of every simulated instance, respectively. These

parameters have been set empirically after the repetition of several experiments and performing a sensitivity analysis. All tests conducted with DWCA have been carried out on an Intel Core i7-7600U laptop, with 2.80 GHz and a 16 GB RAM. On the other hand, tests performed with the other six techniques were run on an Intel Xeon E5 2650 v3 computer, with 2.30 GHz and 32 GB RAM. Java was used as the programming language for all simulations.

Before proceeding with the analysis of the results, it should be highlighted that the main objective of this benchmark is not to find the optimum solution to these problems, but instead to prove that the designed DWCA solver can be adapted to tackle routing problems and hence, to provide empirical evidences of the potential of this meta-heuristic to address other combinatorial optimization problems.

6. Results and Discussion

The discussion begins with Table 2, which shows the optimization results obtained by DWCA, GA, IGA and ESA. Average and best results, standard deviation, and convergence behavior are provided for each (*problem*, *technique*) combination, with the best solution and the best average results across all heuristics highlighted in bold. Relative differences between the best result found and the optima are also depicted. Also is included in the table the mean generation number t_{conv} for which the stopping criterion was met for every technique and problem instance. Furthermore, and for the sake of the completeness of the study, the runtimes rt presented by the DWCA are also presented (in seconds). Specifically, the runtime shown in both computers above described is depicted for each instance (Intel Core i7-7600U - Intel Xeon E5 2650 v3). As previously mentioned, and since the performed experiments have been conducted in different computers, a comparison based on the runtime will not be made in this research. Instead, the convergence behavior will be used for the evaluation of the computational effort.

Different conclusions can be extracted from the obtained outcomes. First, DWCA dominates GA, IGA and ESA in 84.84% of the simulated problem datasets (28 out of 33). Specifically, DWCA scored better results in 84.21% of the TSP datasets, and in 85.71% of the ATSP datasets. Method by method, DWCA outperforms GA in 31 of the 33 simulated cases, and IGA and ESA in 29 of the 33 datasets. These results are supported by the statistical tests discussed later in this section. Important is to highlight the robustness featured by the proposed DWCA solver over the simulated problem datasets. A deeper look at Table 2 reveals that the standard deviations of the results attained by the proposed discrete method are lower than the ones reached by the rest of the algorithms. This noted fact suggests that the quality of the solutions obtained by the DWCA is more stable, promoting DWCA as a more reliable solver in addition to its better performance in terms of optimality. This issue is crucial in case the method is used for solving complex real-world problems for which several runs are not computationally affordable. Finally, analyzing the convergence behavior shown by each technique (quantified by t_{conv}), it is concluded that in this aspect DWCA also outperforms the rest of methods in more than 90% of the simulated cases. This behavior is also a practical advantage, since better results are obtained at a lower number of function evaluations.

Instance		DWCA					GA				IGA				ESA			
		Avg	Best	Std	t_{conv}	rt	Avg	Best	Std	t_{conv}	Avg	Best	Std	t_{conv}	Avg	Best	Std	t_{conv}
Name	Optima	TSP																
Oliver30	420	**420.0**	**420** (+0.0%)	0.0	1.8	0.2 - 0.4	422.8	**420** (+0.0%)	3.4	25.4	421.5	**420** (+0.0%)	2.1	22.7	**420.0**	**420** (+0.0%)	0.0	23.9
Eilon50	425	427.5	**425** (+0.0%)	1.4	7.2	0.8 - 2.7	427.6	426 (+0.2%)	5.8	90.4	**427.0**	**425** (+0.0%)	2.2	90.5	427.4	427 (+0.4%)	1.7	94.3
Eil51	426	428.4	**426** (+0.0%)	2.0	8.0	0.7 - 2.9	440.8	427 (+0.2%)	7.3	88.8	434.4	**426** (+0.0%)	4.5	89.7	431.6	**426** (+0.0%)	2.9	85.9
Berlin52	7542	**7542.0**	**7542** (+0.0%)	0.0	15.7	0.9 - 2.8	**7542.0**	**7542** (+0.0%)	0.0	130.6	**7542.0**	**7542** (+0.0%)	0.0	118.3	**7542.0**	**7542** (+0.0%)	0.0	128.2
St70	675	**678.6**	**675** (+0.0%)	2.2	90.4	2.9 - 9.7	709.8	**675** (+0.0%)	5.7	220.9	690.2	**675** (+0.0%)	9.8	234.7	682.1	**675** (+0.0%)	3.9	216.0
Eilon75	535	545.4	**535** (+0.0%)	4.8	98.8	4.4 - 12.9	565.6	550 (+2.8%)	14.2	269.1	552.4	544 (+1.6%)	7.6	283.1	550.2	545 (+1.8%)	3.9	273.2
Eil76	538	**547.9**	543 (+0.9%)	3.3	116.9	3.7 - 13.0	565.4	545 (+1.3%)	9.8	284.3	557.7	545 (+1.3%)	6.8	276.1	553.7	546 (+1.4%)	4.2	262.8
KroA100	21282	**21348.1**	**21282** (+0.0%)	47.9	473.4	10.7 - 34.1	21812.4	21350 (+0.3%)	420.8	742.7	21731.8	21345 (+0.2%)	340.7	795.4	21481.7	**21282** (+0.0%)	150.1	784.8
KroB100	22140	**22450.7**	22178 (+0.1%)	164.4	605.0	8.9 - 32.6	22687.4	22176 (+0.1%)	407.7	753.4	22712.6	22208 (+0.3%)	312.8	731.6	22602.2	22202 (+0.2%)	210.2	729.8
KroC100	20749	**20934.7**	20769 (+<0.1%)	124.6	555.4	11.7 - 34.7	21510.4	20861 (+0.5%)	390.2	738.2	21298.7	20830 (+0.4%)	290.7	746.1	21170.4	**20749** (+0.0%)	188.7	726.3
KroD100	21294	**21529.6**	21361 (+0.3%)	113.9	526.7	10.4 - 38.2	22184.6	21492 (+0.9%)	405.0	673.4	21696.9	21582 (+1.3%)	408.9	685.0	21726.5	21500 (+0.9%)	156.9	689.4
KroE100	22068	**22246.2**	22130 (+0.3%)	66.1	657.6	10.0 - 38.6	22741.3	22150 (+0.3%)	306.0	813.4	22721.9	22110 (+0.2%)	368.0	800.9	22499.7	22099 (+0.1%)	171.4	791.7
Eil101	629	**645.9**	639 (+1.5%)	2.9	602.2	13.3 - 42.3	673.8	655 (+4.1%)	12.5	613.7	660.7	650 (+3.3%)	7.5	604.3	658.4	650 (+3.3%)	4.4	598.1
Pr107	44303	**44647.1**	44442 (+0.3%)	117.6	602.1	15.7 - 53.7	45619.6	44392 (+0.2%)	1395.4	679.0	44902.5	44428 (+0.2%)	660.3	683.0	44821.5	44413 (+0.2%)	179.3	661.9
Pr124	59030	**59338.9**	**59030** (+0.0%)	163.6	1885.8	29.5 - 86.3	59901.0	**59030** (+0.0%)	562.6	1528.4	59912.8	59072 (+<0.1%)	532.1	1512.4	59593.6	**59030** (+0.0%)	367.8	1446.9
Pr136	96772	**98761.4**	97488 (+0.7%)	741.2	2381.4	35.6 - 136.3	100472.4	98432 (+1.7%)	1225.6	2412.0	99932.7	98532 (+1.8%)	1301.2	2320.9	99858.3	98499 (+1.7%)	655.7	2318.2
Pr144	58537	**58734.6**	**58537** (+0.0%)	167.2	3773.4	45.5 - 145.0	60591.4	58599 (+0.1%)	2342.8	3746.3	58893.0	58581 (+<0.1%)	1012.4	3837.6	58807.3	58574 (+<0.1%)	220.9	3678.4
Pr152	73682	**74202.6**	**73682** (+0.0%)	309.3	4049.5	54.2 - 191.5	75658.3	74520 (+1.1%)	910.8	3782.4	75126.7	74249 (+0.7%)	1005.7	3631.2	74969.5	74172 (+0.6%)	498.9	3853.9
Pr264	49135	**49528.6**	49310 (+0.3%)	1302.7	5963.7	72.3 - 312.7	52499.8	51712 (+5.2%)	932.4	6081.3	52290.0	51653 (+5.1%)	782.7	6208.0	52198.5	51603 (+7.2%)	426.1	6096.4
Name	Optima	ATSP																
br17	39	**39.0**	**39** (+0.0%)	0.0	0.5	0.1 - 0.1	**39.0**	**39** (+0.0%)	0.0	0.4	**39.0**	**39** (+0.0%)	0.0	0.3	**39.0**	**39** (+0.0%)	0.0	0.3
ftv33	1286	**1308.7**	**1286** (+0.0%)	20.5	9.5	0.4 - 1.8	1409.4	1290 (+0.3%)	81.2	15.7	1402.7	**1286** (+0.0%)	99.7	15.3	1322.5	**1286** (+0.0%)	24.5	13.9
ftv35	1473	**1485.8**	**1473** (+0.0%)	9.7	14.1	0.6 - 2.2	1597.2	1490 (+1.1%)	78.4	15.2	1589.0	1498 (+1.6%)	82.1	14.8	**1490.3**	**1473** (+0.0%)	29.5	14.1
ftv38	1530	**1549.0**	**1530** (+0.0%)	16.9	16.8	1.0 - 3.2	1670.4	1565 (+2.2%)	67.4	23.6	1650.1	1560 (+1.9%)	72.1	24.1	1568.8	**1530** (+0.0%)	21.0	23.4
p43	5620	**5620.0**	**5620** (+0.0%)	0.0	8.3	0.9 - 3.0	5625.2	**5620** (+0.0%)	5.4	15.8	**5620.0**	**5620** (+0.0%)	0.0	13.2	**5620.0**	**5620** (+0.0%)	0.0	13.8
ftv44	1613	**1665.0**	**1613** (+0.0%)	24.1	23.8	1.6 - 4.9	1780.0	1649 (+2.2%)	94.7	48.4	1800.3	1645 (+2.0%)	120.7	42.8	1718.9	1645 (+2.0%)	39.2	45.0
ftv47	1776	**1827.8**	**1776** (+0.0%)	32.6	26.0	1.7 - 5.6	1963.1	1820 (+2.4%)	89.6	49.3	1957.4	1822 (+2.2%)	118.4	47.3	1879.8	1795 (+1.0%)	52.7	46.6
ry48p	14422	**14517.8**	14429 (+<0.1%)	53.9	24.3	1.8 - 6.8	14992.1	14545 (+0.8%)	340.7	19.7	14892.0	14530 (+0.7%)	201.8	20.4	14598.0	14485 (+0.4%)	108.7	19.1
ft53	6905	**7199.4**	6971 (+0.9%)	134.4	32.2	2.7 - 10.8	7568.4	7270 (+5.2%)	358.7	66.8	7445.2	7076 (+2.4%)	430.0	65.1	7314.7	6990 (+1.2%)	157.8	65.2
ftv55	1608	**1691.4**	**1608** (+0.0%)	35.8	32.5	3.4 - 12.5	1871.1	1700 (+5.7%)	132.1	75.8	1970.8	1742 (+8.3%)	120.1	69.7	1822.6	1725 (+7.2%)	70.1	75.1
ftv64	1839	**1961.0**	1900 (+3.3%)	33.3	67.1	5.8 - 21.5	2205.7	2014 (+9.5%)	127.4	96.9	2262.1	2080 (+13.1%)	152.1	96.7	2072.3	1955 (+6.3%)	65.0	93.4
ftv70	1950	**2126.2**	2014 (+3.2%)	51.6	88.1	6.7 - 29.1	2315.8	2184 (+12.0%)	140.7	142.7	2351.7	2135 (+9.4%)	134.2	149.1	2312.6	2200 (+12.8%)	67.2	146.7
ft70	38673	**40111.1**	39669 (+2.5%)	252.5	82.2	6.9 - 29.7	40400.7	39407 (+1.9%)	620.4	149.7	40672.4	39241 (+1.4%)	781.8	142.7	40551.4	39650 (+2.5%)	467.2	136.9
kro124p	36230	**39252.8**	37412 (+3.2%)	713.8	468.1	34.0 - 132.3	42250.3	39265 (+8.3%)	1825.4	419.7	42101.9	39099 (+7.9%)	1072.4	429.7	42132.0	40019 (+10.4%)	1250.7	451.0

Table 2: Obtained optimization results using DWCA, GA, IGA, and ESA for TSP and ATSP datasets.

Instance		DWCA					BA				FA				ICA			
		Avg	Best	Std	t_{conv}	rt	Avg	Best	Std	t_{conv}	Avg	Best	Std	t_{conv}	Avg	Best	Std	t_{conv}
Name	Optima	TSP																
Oliver30	420	**420.0**	**420** (+0.0%)	0.0	1.8	0.2 - 0.4	**420.0**	**420** (+0.0%)	0.0	2.1	**420.0**	**420** (+0.0%)	0.0	3.4	**420.0**	**420** (+0.0%)	0.0	3.9
Eilon50	425	427.5	**425** (+0.0%)	1.4	7.2	0.8 - 2.7	427.4	**425** (+0.0%)	1.3	22.8	427.2	**425** (+0.0%)	1.8	17.9	427.9	**425** (+0.0%)	2.1	18.0
Eil51	426	428.4	**426** (+0.0%)	2.0	8.0	0.7 - 2.9	**428.1**	**426** (+0.0%)	1.6	15.3	430.8	**426** (+0.0%)	2.3	18.2	432.3	**426** (+0.0%)	3.1	17.8
Berlin52	7542	**7542.0**	**7542** (+0.0%)	0.0	15.7	2.9 - 2.8	**7542.0**	**7542** (+0.0%)	0.0	20.0	**7542.0**	**7542** (+0.0%)	0.0	23.8	**7542.0**	**7542** (+0.0%)	0.0	24.5
St70	675	**678.6**	**675** (+0.0%)	2.2	90.4	2.9 - 9.7	679.1	**675** (+0.0%)	2.8	72.6	685.3	**675** (+0.0%)	4.0	74.3	684.7	**675** (+0.0%)	3.7	73.8
Eilon75	535	545.4	**535** (+0.0%)	4.8	98.8	4.4 - 12.9	547.4	**535** (+0.0%)	3.9	116.5	**543.6**	**535** (+0.0%)	5.3	173.8	551.7	537 (+0.3%)	6.8	169.8
Eil76	538	**547.9**	543 (+0.9%)	3.3	116.9	3.7 - 13.0	548.1	539 (+0.1%)	3.8	91.5	556.8	543 (+0.9%)	4.9	167.1	557.6	544 (+1.1%)	5.8	168.5
KroA100	21282	**21348.1**	**21282** (+0.0%)	47.9	473.4	10.7 - 34.1	21445.3	**21282** (+0.0%)	116.5	738.8	21483.6	**21282** (+0.0%)	163.7	809.4	21500.3	**21282** (+0.0%)	183.4	820.2
KroB100	22140	**22450.7**	22178 (+0.1%)	164.4	605.0	8.9 - 32.6	22506.4	**22140** (+0.0%)	221.3	461.0	22604.8	22183 (+0.1%)	243.9	820.3	22599.7	22180 (+0.1%)	244.9	816.8
KroC100	20749	**20934.7**	20769 (+<0.1%)	124.6	555.4	11.7 - 34.7	21050.0	**20749** (+0.0%)	164.7	872.2	21096.3	20756 (+<0.1%)	148.3	818.6	21103.9	20756 (+<0.1%)	161.1	820.0
KroD100	21294	**21529.6**	21361 (+0.3%)	113.9	526.7	10.4 - 38.2	21593.4	**21294** (+0.0%)	141.6	600.3	21683.8	21408 (+0.5%)	163.7	843.5	21666.8	21399 (+0.4%)	174.0	840.2
KroE100	22068	**22246.2**	22130 (+0.3%)	66.1	657.6	10.0 - 38.6	22349.6	**22068** (+0.0%)	169.6	602.9	22413.0	22079 (+<0.1%)	183.0	883.1	22453.3	22083 (+<0.1%)	196.9	881.3
Eil101	629	**645.9**	639 (+1.5%)	2.9	602.2	13.3 - 42.3	646.4	634 (+0.7%)	4.9	512.7	659.0	643 (+2.2%)	8.1	618.2	663.8	644 (+2.3%)	9.6	622.1
Pr107	44303	**44647.1**	44442 (+0.3%)	117.6	602.1	15.7 - 53.7	44793.8	**44303** (+0.0%)	232.4	679.0	44790.4	**44303** (+0.0%)	227.3	725.0	44803.3	**44303** (+0.0%)	302.7	721.4
Pr124	59030	**59338.9**	**59030** (+0.0%)	163.6	1885.8	29.5 - 86.3	59412.1	**59030** (+0.0%)	265.9	1602.5	59404.3	**59030** (+0.0%)	257.9	1603.0	59436.9	**59030** (+0.0%)	299.4	1612.4
Pr136	96772	**98761.4**	97488 (+0.7%)	741.2	2381.4	35.6 - 136.3	99351.2	97547 (+0.8%)	707.2	2866.6	99683.7	97716 (+0.9%)	831.3	2769.9	99583.7	97736 (+0.9%)	848.9	2775.2
Pr144	58537	**58734.6**	**58537** (+0.0%)	167.2	3773.4	42.2- 145.0	58876.2	**58537** (+0.0%)	295.6	4361.1	58993.3	58546 (+<0.1%)	300.1	4131.1	59070.9	58563 (+<0.1%)	323.0	4122.7
Pr152	73682	**74202.6**	**73682** (+0.0%)	309.3	4049.5	54.2 - 191.5	74676.9	73921 (+0.3%)	426.5	4853.1	74934.3	74033 (+0.4%)	483.7	4811.1	74886.7	74052 (+0.5%)	513.9	4794.8
Pr264	49135	**49528.6**	48310 (+0.3%)	1302.7	5963.7	72.3 - 312.7	50908.3	49756 (+1.2%)	887.0	6375.4	51837.0	50491 (+2.7%)	760.8	6733.0	51943.6	50553 (+2.8%)	863.7	6721.0
Name	Optima	ATSP																
br17	39	**39.0**	**39** (+0.0%)	0.0	0.5	0.1 - 0.1	**39.0**	**39** (+0.0%)	0.0	0.3	**39.0**	**39** (+0.0%)	0.0	0.3	**39.0**	**39** (+0.0%)	0.0	0.3
ftv33	1286	**1308.7**	**1286** (+0.0%)	20.5	9.5	0.4 - 1.8	1318.1	**1286** (+0.0%)	25.7	14.6	1320.9	**1286** (+0.0%)	21.9	14.7	1324.6	**1286** (+0.0%)	28.3	15.1
ftv35	1473	**1485.8**	**1473** (+0.0%)	9.7	14.1	0.6 - 2.2	1493.7	**1473** (+0.0%)	8.0	12.7	1498.8	**1473** (+0.0%)	10.4	16.2	1490.6	**1473** (+0.0%)	11.9	15.2
ftv38	1530	**1549.0**	**1530** (+0.0%)	16.9	16.8	1.0 - 3.2	1562.0	**1530** (+0.0%)	13.7	25.5	1560.4	**1530** (+0.0%)	14.6	26.5	1565.6	**1530** (+0.0%)	15.8	27.1
p43	5620	**5620.0**	**5620** (+0.0%)	0.0	8.3	0.9 - 3.0	**5620.0**	**5620** (+0.0%)	0.0	11.8	**5620.0**	**5620** (+0.0%)	0.0	15.3	**5620.0**	**5620** (+0.0%)	0.0	14.7
ftv44	1613	**1665.0**	**1613** (+0.0%)	24.1	23.8	1.6 - 4.9	1683.7	**1613** (+0.0%)	27.2	41.8	1690.8	1620 (+0.4%)	32.3	47.2	1694.3	1622 (+0.5%)	54.0	46.9
ftv47	1776	**1827.8**	**1776** (+0.0%)	32.6	26.0	1.7 - 5.6	1863.6	1796 (+1.1%)	39.3	47.8	1858.3	1795 (+1.0%)	63.4	46.8	1873.0	1799 (+1.2%)	70.1	45.9
ry48p	14422	**14517.8**	14429 (+<0.1%)	53.9	24.3	1.8 - 6.8	14544.8	**14422** (+0.0%)	79.7	42.1	14694.4	14453 (+0.2%)	94.7	54.0	14689.8	14463 (+0.2%)	73.6	53.4
ft53	6905	**7199.4**	6971 (+0.9%)	134.4	32.2	2.7 - 10.8	7294.1	7001 (+1.3%)	196.9	61.6	7302.0	6993 (+1.2%)	186.4	70.8	7320.1	7002 (+1.4%)	200.3	69.4
ftv55	1608	**1691.4**	**1608** (+0.0%)	35.8	32.5	3.4 - 12.5	1737.5	**1608** (+0.0%)	50.5	71.8	1790.6	1628 (+1.2%)	64.0	76.0	1801.4	1630 (+1.3%)	83.0	78.4
ftv64	1839	**1961.0**	1900 (+3.3%)	33.3	67.1	5.8 - 21.5	1999.2	1879 (+2.1%)	68.2	89.8	2041.6	1903 (+3.4%)	73.4	99.4	2040.8	1900 (+3.3%)	81.3	99.8
ftv70	1950	**2126.2**	2014 (+3.2%)	51.6	88.1	6.7 - 29.1	2233.2	2111 (+8.2%)	48.8	134.9	2290.8	2173 (+11.4%)	60.0	155.9	2322.6	2167 (+11.1%)	63.3	152.6
ft70	38673	**40111.1**	39669 (+2.5%)	252.5	82.2	6.9 - 29.7	40309.7	39901 (+3.1%)	131.5	136.7	40694.8	39668 (+2.5%)	494.6	142.8	40699.7	39660 (+2.5%)	534.9	140.0
kro124p	36230	39252.8	37412 (+3.2%)	713.8	468.1	34.0 - 132.3	**39213.7**	37538 (+3.6%)	438.5	451.0	41637.5	39438 (+8.8%)	1094.7	461.5	41608.3	39400 (+8.7%)	1116.8	456.6

Table 3: Obtained optimization results using DWCA, BA, FA, and ICA for TSP and ATSP datasets.

The discussion follows with Table 3, where the results obtained by ICA, BA and FA are shown along with those achieved by the proposed DWCA. Similar conclusions can be drawn from this second result set; in this case, DWCA also outstands as the solver with best results. Specifically, it shows the best performance in 73.68% of the TSP datasets (14 out of 19), with significantly worse results only in 3 datasets. In regards to ATSP datasets, DWCA obtained better results in 78.57% of the cases, showing the same performance in 2 datasets, and worse performance just in one single case. Overall, DWCA outperforms the rest of the alternatives in 75.75% of the datasets (25 out

of 33), being surpassed only in 12.12% of the cases. When comparing among every pair of techniques, DWCA outperformed BA in 78.78% of the datasets, FA in 81.81% of the cases, and ICA in 29 of the 33 datasets. On the other hand, DWCA has been surpassed by BA only in 3 datasets, and by FA just in 2 of the cases. Furthermore, the ICA never occurred to outperform DWCA. Conclusions about the robustness and the convergence of DWCA hold as in the previous experimentation.

TSP		ATSP	
Algorithm	Ranking	Algorithm	Ranking
DWCA	1.5789	DWCA	1.4643
BA	2.3684	BA	2.4643
ESA	4.2632	ESA	4.2857
GA	6.5526	GA	6.1429
IGA	5.6316	IGA	5.6786
FA	3.5000	FA	3.6786
ICA	4.1053	ICA	4.2857

Table 4: Average rankings returned by the Friedman's non-parametric test for the results' quality.

Following the guidelines in [19], two different tests have been carried out to resolve the statistical relevance of the reported performance gaps. To begin with, the Friedman's non-parametric test for multiple comparison allows proving if there are significant differences in the results obtained by all reported methods. Table 4 displays the mean ranking returned by this nonparametric test for each of the compared algorithms and problems (the lower the rank, the better the performance). Thus, for the TSP the Friedman statistic (distributed according to χ^2 with 6 degrees of freedom) was equal to 73.415. Furthermore, the confidence interval has been set to 99%, being 16.812 the critical point in a χ^2 distribution with 6 degrees of freedom. Since $73.415 > 16.812$, it can be concluded that there are significant differences among the results, thus DWCA can be regarded as the method having the lowest rank. Regarding ATSP datasets, and taking into account also the results shown in Table 4, the Friedman statistic test for this problem family is 49.393. Since $49.39 > 16.812$, the same conclusion follows as for the TSP problem portfolio.

The second statistical test is the Holm's post-hoc test. For correctly conducting this test, DWCA has been set as the control algorithm. Table 5 gathers the unadjusted and adjusted p-values obtained through the application of Holm's post-hoc procedure. From these p-values it can be concluded that DWCA is significantly better than GA, IGA, ESA, FA, and ICA for both TSP and ATSP at a 95% confidence level. Finally, the DWCA is better than the BA for both TSP and ATSP, but not significantly from the statistical point of view.

In order to delve further into the comparison among DWCA and BA, an additional statistical analysis has been made in terms of convergence behavior. The analysis aims at exploring whether the lack of significance in terms of optimality between both solvers spans to their convergence behavior, in other words, whether both solvers perform statistically similarly in terms of convergence speed and optimality. To this end, both non-parametric Friedman and post-hoc Holm's tests have been conducted over the recorded values of t_{conv} for every problem instance and DWCA and BA. In this sense, the obtained Friedman statistic (Table 6.a) has been 25.136. Under a confidence interval of 99% (with 9.210 as the critical point in a χ^2 distribution with 2 degrees of

TSP			ATSP		
Algorithm	Unadjusted p	Adjusted p	Algorithm	Unadjusted p	Adjusted p
GA	0	0	GA	0	0
IGA	0	0	IGA	0	0
ESA	0.000128	0.000513	ESA	0.000549	0.002197
ICA	0.000313	0.000938	ICA	0.000549	0.002197
FA	0.006127	0.012253	FA	0.006689	0.013378
BA	0.259992	0.259992	BA	0.220671	0.220671

Table 5: Unadjusted and adjusted p-values obtained as a result of the application of Holm's post-hoc procedure using DWCA as the control algorithm.

freedom), it can be said that there are significant differences, thus the DWCA is the technique with the lowest rank. This interesting result is also buttressed by the results depicted in Table 6.b, where the Holm's test proves that DWCA is significantly better than both BA and FA optimizers (all p-values are lower than 0.05).

Algorithm	Ranking
DWCA	1.4242
BA	1.9242
FA	2.6515

(a)

Algorithm	Unadjusted p	Adjusted p
FA	0.000001	0.000001
BA	0.042254	0.042254

(b)

Table 6: (a) Average rankings returned by the Friedman's non-parametric test for the convergence behavior analysis; (b) unadjusted and adjusted p-values obtained as a result of the application of Holm's post-hoc procedure using DWCA as the control algorithm for the convergence behavior analysis.

Finally, and with the aim of seeking completeness of this research, the convergence behavior of all seven methods are directly compared in the following Figure 5 for the TSP, and Figure 6 for the ATSP. These figures support the analysis described in this section, which concludes that the DWCA presents a promising performance also in this aspect.

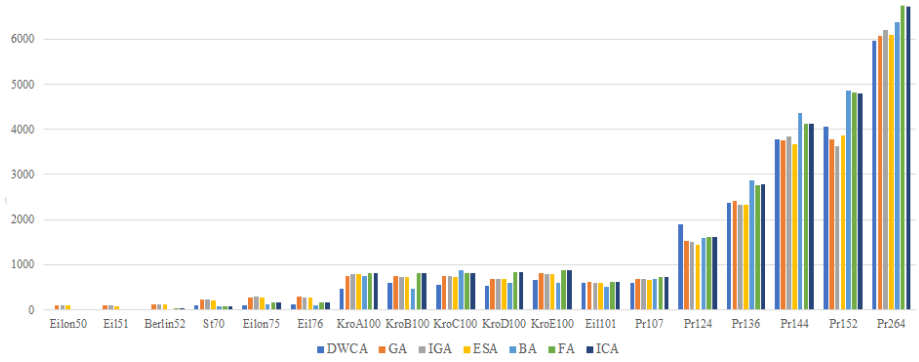


Figure 5: Convergence behavior shown by all the methods for the TSP.

As a final conclusion, all the above results evince that the proposed DWCA solver outperforms the other six methods in terms of quality, robustness, and convergence

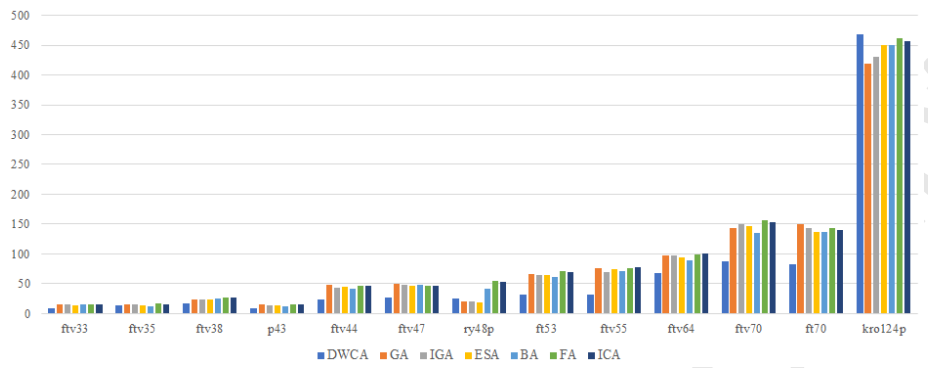


Figure 6: Convergence behavior shown by all the approaches for the ATSP.

of the routes found for every problem instance, dominance supported by two different statistical tests in almost all simulated cases. Thus, the proposed DWCA can be deemed as a promising search method for solving practical TSP and ATSP scenarios.

7. Conclusions and Future Research Lines

This manuscript has presented a discrete version of the so-called Water Cycle Algorithm suited for solving two well-known optimization problems: the symmetric and Asymmetric Traveling Salesman Problem (resp. TSP and ATSP). The proposed DWCA solver maintains the inspiration in the hydrological phenomena underlying the original WCA, yet incorporates novel ingredients for efficiently undertaking the considered routing problems: 1) the use of the Hamming distance as the measure to compute the difference between routes found during the search process; 2) an adaptive modification of the movement function depending on the estimated *inclination* of the stream/river with respect to its assigned river/sea; and 3) an insertion-based mutation operator that emulates the evaporation and raining process in the discrete solution space spanned by a permutation encoding strategy. To empirically assess the performance of the proposed approach, 33 datasets of the renowned TSPLIB library have been utilized as a benchmark baseline, over which the performance of six different meta-heuristics from the literature (ESA, GA, IGA, BA, FA and ICA) and that of DWCA have been compared to each other in terms of both convergence rate and optimality. In overall, the developed discrete WCA (DWCA) has shown better performance than the rest of solvers, not only in regards to the quality of the produced results, but also in the speed at which they are obtained and its statistical robustness and stability over different executions of every method. The significance of these conclusions has been positively reinforced by both the non-parametric Friedman test and the post-hoc Holm's test.

Several research lines stem directly from the findings reported in this work. Other combinatorial optimization problems could benefit from the adaptations to the original WCA reported in this study; in this context, discrete problems such as the multi-dimensional bin packing problem or the selective pick-up and delivery problem will be under study due to the plethora of studies reported in the literature dealing with

meta-heuristics adapted to such cases. Likewise, efforts will be invested in the longer term towards furnishing a solid comparison of the proposed approach to different exact methods and commercial solvers available for the TSP/ATSP, using alternative scores and metrics such as the number of explored solutions or the convergence behavior. Although this family of solvers are different in concept from the designed DWCA, this comparison will be of utmost utility for gaining insight into the potentiality of DWCA for its use in practical scenarios and commercial deployments. Furthermore, in this work, the DWCA has been compared with search based techniques with similar philosophy and characteristics. Additional comparisons are planned for the near future with methods of different nature, such as constructive heuristics (e.g. ACO).

Acknowledgements

This work has been supported by the Basque Government through the ELKARTEK program (ref. KK-2015/0000080 and the BID3ABI project), as well as by the H2020 programme of the European Commission (grant no. 691735).

References

[1] 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 (1-2), 187–206.

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

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

[4] Anbuudayasankar, S., Ganesh, K., Mohapatra, S., 2014. Survey of methodologies for tsp and vrp. In: Models for Practical Routing Problems in Logistics. Springer, pp. 11–42.

[5] 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.

[6] 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.

[7] Baghipour, R., Hosseini, S., Boor, Z., 2014. A water cycle algorithm for optimal allocation of dgs in distribution systems considering environmental profits. International Journal of Mechatronics, Electrical and Computer Technology 4 (11), 430–454.

[8] Bell, J. E., McMullen, P. R., 2004. Ant colony optimization techniques for the vehicle routing problem. Advanced engineering informatics 18 (1), 41–48.

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

[10] Bräysy, O., Gendreau, M., 2005. Vehicle routing problem with time windows, part i: Route construction and local search algorithms. Transportation science 39 (1), 104–118.

[11] Cantú-Paz, E., 2001. Migration policies, selection pressure, and parallel evolutionary algorithms. Journal of heuristics 7 (4), 311–334.

[12] Chawla, M., Duhan, M., 2015. Bat algorithm: a survey of the state-of-the-art. Applied Artificial Intelligence 29 (6), 617–634.

[13] 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 (12), 14439–14450.

[14] Christofides, N., 1976. The vehicle routing problem. RAIRO-Operations Research-Recherche Opérationnelle 10 (V1), 55–70.

[15] Crawford, B., Soto, R., Riquelme-Leiva, M., Peña, C., Torres-Rojas, C., Johnson, F., Paredes, F., 2015. Modified binary firefly algorithms with different transfer functions for solving set covering problems. In: Software Engineering in Intelligent Systems. Springer, pp. 307–315.

[16] Dantzig, G., Fulkerson, R., Johnson, S., 1954. Solution of a large-scale traveling-salesman problem. Journal of the operations research society of America 2 (4), 393–410.

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

[18] De Jong, K., 1975. Analysis of the behavior of a class of genetic adaptive systems. Ph.D. thesis, University of Michigan, Michigan, USA.

[19] 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 (1), 3–18.

[20] Eskandar, H., Sadollah, A., Bahreininejad, Ardeshir, Hamdi, M., 2012. Water cycle algorithm a novel metaheuristic optimization method for solving constrained engineering optimization problems. Applied Soft Computing 110 (111), 151–166.

[21] Fiechter, C.-N., 1994. A parallel tabu search algorithm for large traveling salesman problems. Discrete Applied Mathematics 51 (3), 243–267.

[22] Fister, I., Fister Jr, I., Yang, X.-S., Brest, J., 2013. A comprehensive review of firefly algorithms. Swarm and Evolutionary Computation.

[23] 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. Springer, pp. 347–360.

[24] Gao, K., Duan, P., Su, R., Li, J., 2017. Bi-objective water cycle algorithm for solving remanufacturing rescheduling problem. In: Shi, Y., Tan, K. C., Zhang, M., Tang, K., Li, X., Zhang, Q., Tan, Y., Middendorf, M., Jin, Y. (Eds.), Simulated Evolution and Learning. Springer International Publishing, Cham, pp. 671–683.

[25] Gao, K., Zhang, Y., Sadollah, A. Lentzakis, A., Su, R., 2017. Jaya, harmony search, and water cycle algorithms for solving large-scale real-life urban traffic light scheduling problem. Swarm and Evolutionary Computation https://doi.org/10.1016/j.swevo.2017.05.002.

[26] Ghaffarzadeh, N., 2015. Water cycle algorithm based power system stabilizer robust design for power systems. Journal of Electrical Engineering 66 (2), 91–96.

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

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

[29] 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. Lawrence Erlbaum, New Jersey (160-168), pp. 160–168.

[30] Gülcü, Ş., Mahi, M., Baykan, Ö. K., Kodaz, H., 2016. A parallel cooperative hybrid method based on ant colony optimization and 3-opt algorithm for solving traveling salesman problem. Soft Computing, 1–17.

[31] Guney, K., Basbug, S., 2014. A quantized water cycle optimization algorithm for antenna array synthesis by using digital phase shifters. International Journal of RF and Microwave Computer-Aided Engineering 25 (1), 21–29.

[32] Guo, Y., Li, B., 2018. Novel method for parameter estimation of newtons rings based on cfrft and er-wca. Signal Processing 144, 118–126.

[33] Haddad, O., Moravej, M., Loiciga, H., 2014. Application of the water cycle algorithm to the optimal operation of reservoir systems. Journal of Irrigation and Drainage Engineering 141 (5), 04014064.

[34] Heidari, A., Abbaspour, R., Rezaee Jordehi, A., 2017. Gaussian bare-bones water cycle algorithm for optimal reactive power dispatch in electrical power systems. Applied Soft Computing 57, 657–671.

[35] Hoffman, K. L., Padberg, M., Rinaldi, G., 2013. Traveling salesman problem. In: Encyclopedia of operations research and management science. Springer, pp. 1573–1578.

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

[37] 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, 295–312.

[38] Jordehi, A. R., 2016. Optimal allocation of facts devices for static security enhancement in power systems via imperialistic competitive algorithm (ica). Applied Soft Computing 48, 317–328.

[39] Karimi, N., Davoudpour, H., 2017. A knowledge-based approach for multi-factory production systems. Computers & Operations Research 77, 72–85.

[40] Kashan, A. H., 2009. League championship algorithm: a new algorithm for numerical function optimization. In: Soft Computing and Pattern Recognition, 2009. SOCPAR'09. International Conference of. IEEE, pp. 43–48.

[41] Kennedy, J., Eberhart, R., et al., 1995. Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks. Vol. 4. Perth, Australia, pp. 1942–1948.

[42] Kirkpatrick, S., Gellat, C., Vecchi, M., 1983. Optimization by simulated annealing. science 220 (4598), 671–680.

[43] Knox, J., 1994. Tabu search performance on the symmetric traveling salesman problem. Computers & Operations Research 21 (8), 867–876.

[44] Kong, Y., Mei, Y., Li, W., Wang, X., Ben, Y., 2017. An enhanced water cycle algorithm for optimization of multi-reservoir systems. In: IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS).

[45] Laporte, G., 1992. The traveling salesman problem: An overview of exact and approximate algorithms. European Journal of Operational Research 59 (2), 231–247.

[46] Laporte, G., 1992. The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operational Research 59 (3), 345–358.

[47] 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 (2), 129–170.

[48] Lawler, E. L., Lenstra, J. K., Kan, A. R., Shmoys, D. B., 1985. The traveling salesman problem: a guided tour of combinatorial optimization. Vol. 3. Wiley New York.

[49] Li, H., Alidaee, B., 2016. Tabu search for solving the black-and-white travelling salesman problem. Journal of the Operational Research Society 67 (8), 1061–1079.

[50] 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 (7), 1277–1281.

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

[52] 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.

[53] 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 (1), 59–84.

[54] Marinakis, Y., Marinaki, M., Migdalas, A., 2015. Adaptive tunning of all parameters in a multi-swarm particle swarm optimization algorithm: An application to the probabilistic traveling salesman problem. In: Optimization, Control, and Applications in the Information Age. Springer, pp. 187–207.

[55] Mavrovouniotis, M., Yang, S., 2013. Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors. Applied Soft Computing 13 (10), 4023–4037.

[56] Mirjalili, S., May 2016. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Computing and Applications 27 (4), 1053–1073.
URL https://doi.org/10.1007/s00521-015-1920-1

[57] Mirjalili, S., Lewis, A., 2013. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. Swarm and Evolutionary Computation 9, 1–14.

[58] Mirjalili, S., Mirjalili, S. M., Yang, X.-S., 2014. Binary bat algorithm. Neural Computing and Applications 25 (3-4), 663–681.

[59] Oliver, I., Smith, D., Holland, J. R., 1987. Study of permutation crossover operators on the traveling salesman problem. In: Genetic algorithms and their applications: proceedings of the second International Conference on Genetic Algorithms: July 28-31, 1987 at the Massachusetts Institute of Technology, Cambridge, MA. Hillsdale, NJ: L. Erlhaum Associates, 1987.

[60] Osaba, E., Carballedo, R., Diaz, F., Onieva, E., Masegosa, A., Perallos, A., 2018. Good practice proposal for the implementation, presentation, and comparison of metaheuristics for solving routing problems. Neurocomputing 271, 2–8.

[61] Osaba, E., Yang, X.-S., Diaz, F., Lopez-Garcia, P., Carballedo, R., 2016. An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems. Engineering Applications of Artificial Intelligence 48, 59–71.

[62] Ouaarab, A., Ahiod, B., Yang, X.-S., 2014. Discrete cuckoo search algorithm for the travelling salesman problem. Neural Computing and Applications 24 (7-8), 1659–1669.

[63] Pahnehkolaei, S., Alfi, A., Sadollah, A., Kim, J., 2017. Gradient-based water cycle algorithm with evaporation rate applied to chaos suppression. Applied Soft Computing 53, 420–440.

[64] Pettey, C. B., Leuze, M. R., Grefenstette, J. J., 1987. Parallel genetic algorithm. In: Genetic algorithms and their applications: proceedings of the second International Conference on Genetic Algorithms: July 28-31, 1987 at the Massachusetts Institute of Technology, Cambridge, MA. Hillsdale, NJ: L. Erlhaum Associates, 1987.

[65] Praepanichawat, C., Khompatraporn, C., Jaturanonda, C., Chotyakul, C., 2015. Water cycle and artificial bee colony based algorithms for optimal order allocation problem with mixed quantity discount scheme. Lecture Notes in Electrical Engineering 349, 229–239.

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

[67] Rezk, H., Fathy, A., 2017. A novel optimal parameters identification of triple-junction solar cell based on a recently meta-heuristic water cycle algorithm. Solar Energy 157 (15), 778–791.

[68] 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.

[69] Saremi, S., Mirjalili, S., Lewis, A., 2015. How important is a transfer function in discrete heuristic algorithms. Neural Computing and Applications 26 (3), 625–640.

[70] Soto, R., Crawford, B., Olivares, R., Barraza, J., Figueroa, I., Johnson, F., Paredes, F., Olguín, E., 2017. Solving the non-unicost set covering problem by using cuckoo search and black hole optimization. Natural Computing 16 (2), 213–229.

[71] Syswerda, G., 1991. Schedule optimization using genetic algorithms. Handbook of genetic algorithms, 332–349.

[72] Taillard, E. D., 2001. Comparison of non-deterministic iterative methods. In: MIC2001, 4th Metaheuristics Internat. Conf. pp. 16–20.

[73] 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.

[74] 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 (3), 806–822.

[75] Wang, J., Ersoy, O. K., He, M., Wang, F., 2016. Multi-offspring genetic algorithm and its application to the traveling salesman problem. Applied Soft Computing 43, 415–423.

[76] Wang, J., Liu, S., 2018. Novel binary encoding water cycle algorithm for solving bayesian network structures learning problem. Knowledge-Based Systems.

[77] Yang, X.-S., 2009. Firefly algorithms for multimodal optimization. In: Stochastic algorithms: foundations and applications. Springer, pp. 169–178.

[78] Yang, X.-S., 2010. A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization. Springer, pp. 65–74.

[79] Yang, X.-S., Deb, S., 2009. Cuckoo search via lévy flights. In: Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on. IEEE, pp. 210–214.

[80] Yang, X.-S., He, X., 2013. Bat algorithm: literature review and applications. International Journal of Bio-Inspired Computation 5 (3), 141–149.

[81] Yip, P. P., Pao, Y.-H., 1995. Combinatorial optimization with use of guided evolutionary simulated annealing. IEEE Transactions on Neural Networks 6 (2), 290–295.

[82] Yousefikhoshbakht, M., Sedighpour, M., 2013. New imperialist competitive algorithm to solve the travelling salesman problem. International Journal of Computer Mathematics 90 (7), 1495–1505.

[83] Yu, B., Yang, Z.-Z., Yao, B., 2009. An improved ant colony optimization for vehicle routing problem. European journal of operational research 196 (1), 171–176.

[84] Zaji, A. H., Bonakdari, H., Khodashenas, S. R., Shamshirband, S., 2016. Firefly optimization algorithm effect on support vector regression prediction improvement of a modified labyrinth side weir's discharge coefficient. Applied Mathematics and Computation 274, 14–19.

[85] Zhan, S.-h., Lin, J., Zhang, Z.-j., Zhong, Y.-w., 2016. List-based simulated annealing algorithm for traveling salesman problem. Computational intelligence and neuroscience 2016, 8.

[86] 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 (7), 1165–1169.

Highlights

- An improved discrete Water Cycle Algorithm is presented for the TSP and ATSP.
- This version includes inclination feature, enhancing exploration and exploitation.
- 33 datasets of the TSP/ATSP have been used for the experimentation.
- Results have been compared with six different techniques.
- Friedman's and Holm's post hoc statistical tests have been conducted.

Figure 1
Click here to download high resolution image

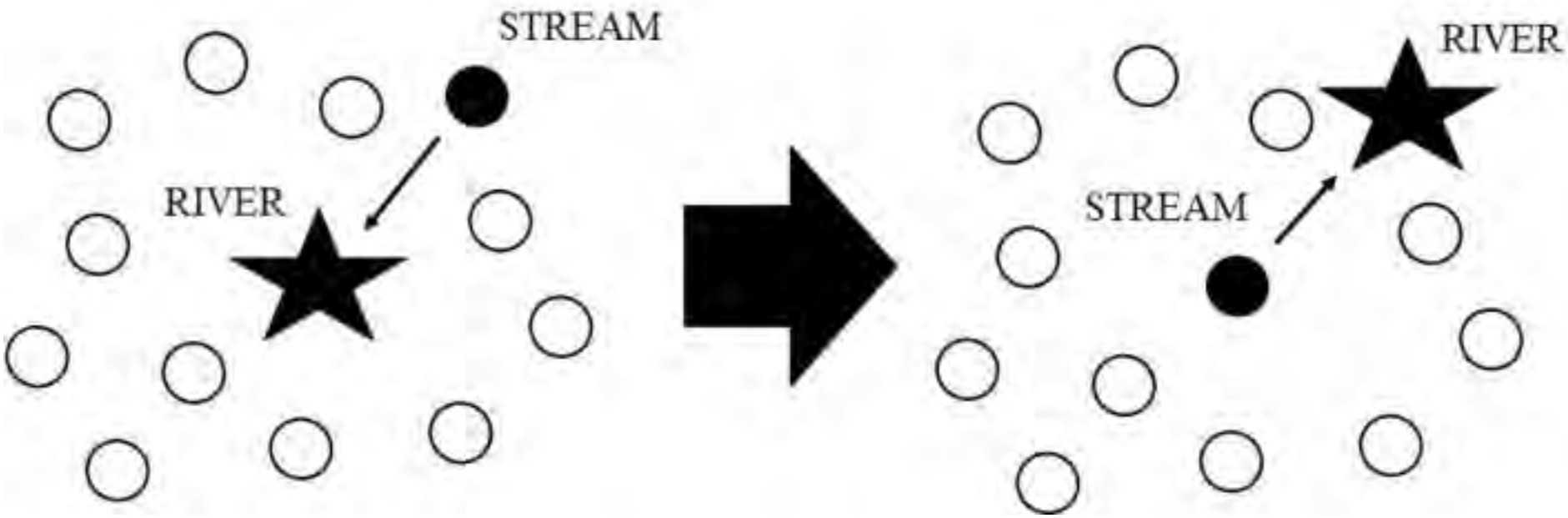
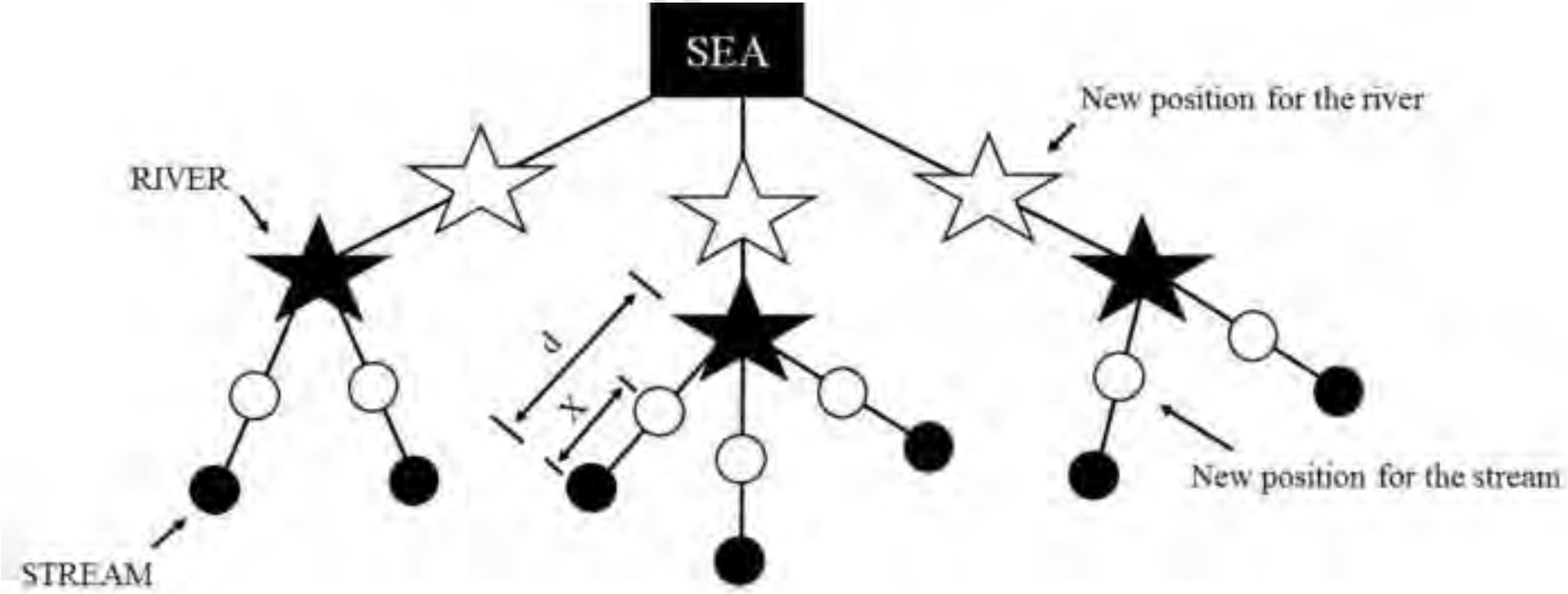


Figure 2
Click here to download high resolution image



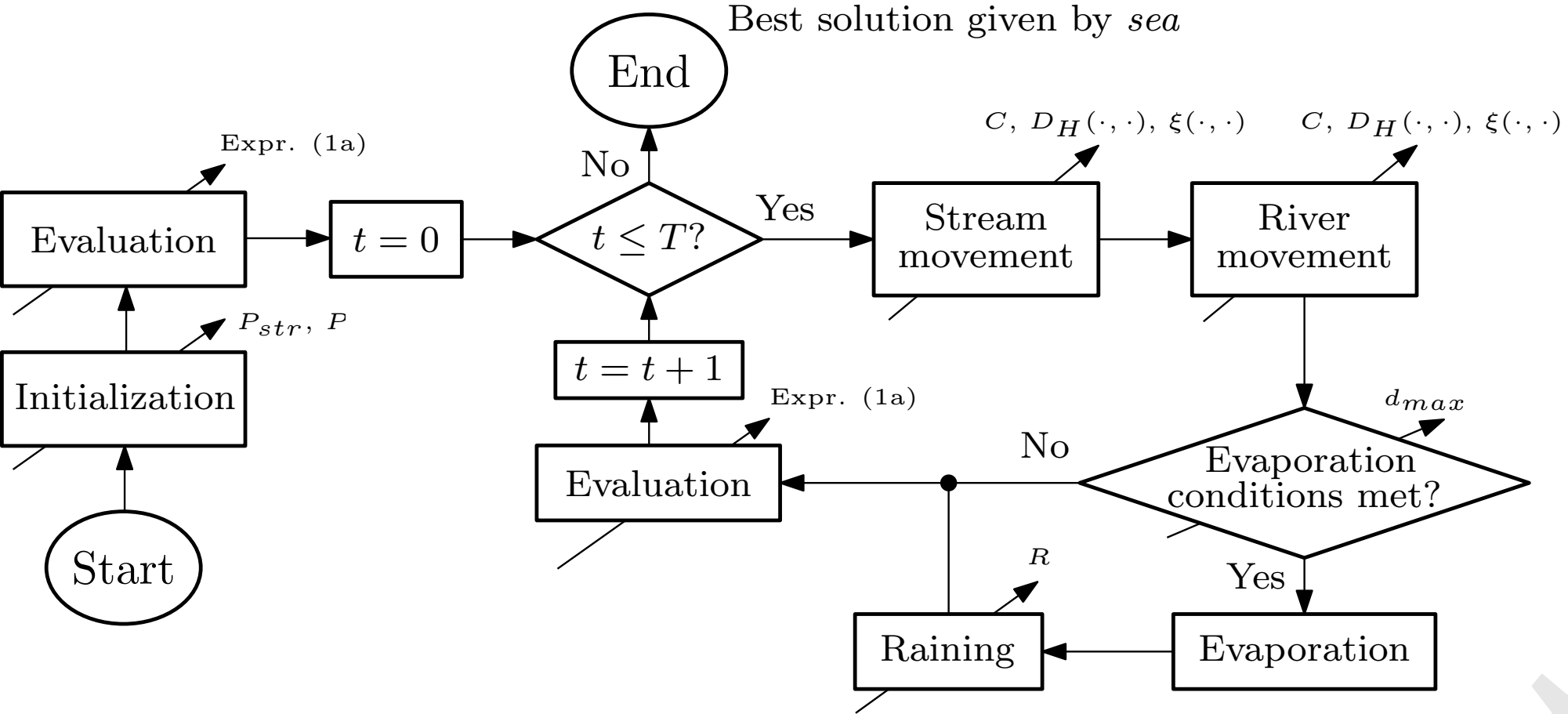

Best solution given by *sea*
End
Expr. (1a)
Evaluation
$t = 0$
No
$t \leq T$?
Yes
C, $D_H(\cdot,\cdot)$, $\xi(\cdot,\cdot)$
Stream movement
C, $D_H(\cdot,\cdot)$, $\xi(\cdot,\cdot)$
River movement
P_{str}, P
Initialization
$t = t + 1$
Expr. (1a)
Evaluation
d_{max}
No
Evaporation conditions met?
Start
R
Yes
Raining
Evaporation

Figure 4
Click here to download high resolution image

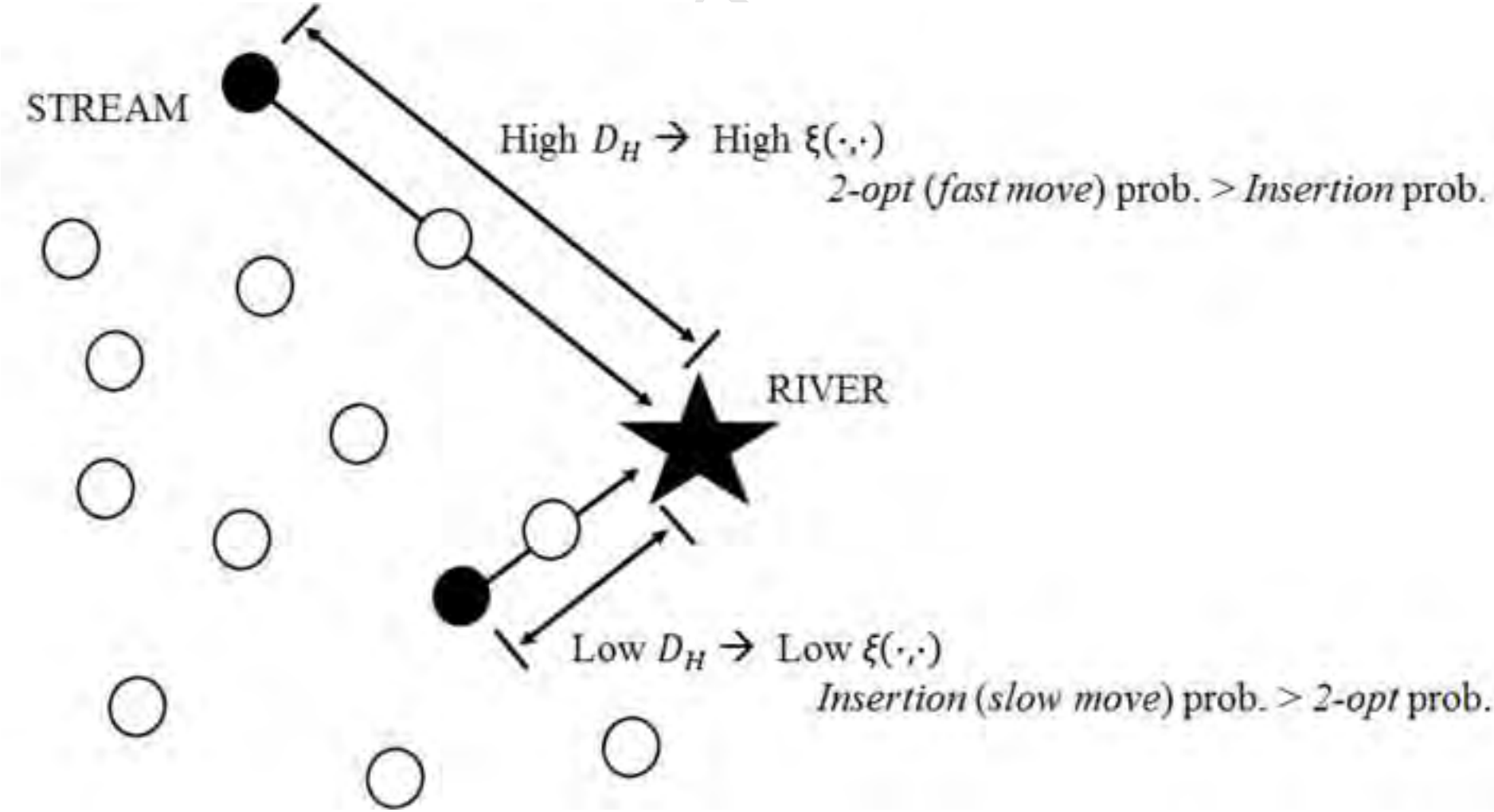


Figure 5

Click here to download high resolution image

Figure 6
Click here to download high resolution image