The AddACO: A bio-inspired modified version of the ant colony optimization algorithm to solve travel salesman problems

Highlights

The AddACO: a bio-inspired modified version of the Ant Colony Optimization Algorithm to solve Travel Salesman Problems

Marco Scianna
Department of Mathematical Sciences, Politecnico di Torino

Corso Duca degli Abruzzi 24, 10129, Torino, Italy

- The work presents the AddACO, a bio-inspired modification of the classical Dorigo's Ant Algorithm to solve Travel Salesman Problems
- The proposed algorithm is based on an additive version of the ant decisional rule
- Numerical simulations assess the performance of the method in the case of selected benchmark distributions of cities of various scales
- Performance comparison with the canonical Ant-Cycle Ant System is proposed
- Analytical results on the asymptotic convergence of the algorithm are finally presented

The AddACO: a bio-inspired modified version of the Ant Colony Optimization Algorithm to solve Travel Salesman Problems

Marco Scianna

Department of Mathematical Sciences, Politecnico di Torino

Corso Duca degli Abruzzi 24, 10129, Torino, Italy

Abstract

The Travel Salesman Problem (TSP) consists in finding the minimal-length closed tour that connects the entire group of nodes of a given graph. We propose to solve such a combinatorial optimization problem with the AddACO algorithm: it is a version of the Ant Colony Optimization method that is characterized by a modified probabilistic law at the basis of the exploratory movement of the artificial insects. In particular, the ant decisional rule is here set to amount in a linear convex combination of *competing* behavioral stimuli and has therefore an *additive* form (hence the name of our algorithm), rather than the canonical multiplicative one. The AddACO intends to address two conceptual shortcomings that characterize classical ACO methods: (i) the population of artificial insects is in principle allowed to simultaneously minimize/maximize all migratory guidance cues (which is in implausible from a biological/ecological point of view) and (ii) a given edge of the graph has a null probability to be explored if at least one of the movement trait is therein equal to zero, i.e., regardless the intensity of the others (this in principle reduces the exploratory potential of the ant colony). Three possible variants of our method are then specified: the AddACO-V1, which includes pheromone trail and visibility as insect decisional variables, and the AddACO-V2 and the AddACO-V3, which in turn add random effects and inertia, respectively, to the two classical migratory stimuli. The three versions of our algorithm are tested on benchmark middle-scale TPS instances, in order to assess their performance and to find their optimal parameter setting. The best performing variant is finally applied to large-scale TSPs, compared to the naive Ant-Cycle Ant System, proposed by Dorigo and colleagues, and evaluated in terms of quality of the solutions, computational time, and convergence speed. The aim is in fact to show that the proposed transition probability, as long as its conceptual advantages, is competitive from a performance perspective, i.e., if it does not reduce the exploratory capacity of the ant population w.r.t. the canonical one (at least in the case of selected TSPs). A theoretical study of the asymptotic behavior of the AddACO is given in the appendix of the work, whose conclusive section contains some hints for further improvements of our algorithm, also in the perspective of its application to other optimization problems.

Keywords: travel salesman problem, ant colony optimization, approximated algorithms, asymptotic convergence

1. Introduction

Optimization plays a relevant role both in the industrial and in the scientific world: generally speaking, its focus is to find the best possibile solution/s, in terms of minimization of a so-called *cost/objective* function, of a given problem in a reasonable time limit [14]. For instance, optimization in the field of model fitting amounts in the search of the characteristic parameters of a theoretical curve that give the best reproduction of a set of experimental data. In particular, the result is obtained by minimization of an error function that measures the discrepancy between predicted and empirical values.

Optimization problems can be divided in two classes: *numeric* or *combinatorial*, depending on whether the set of candidate solutions is continuous or discrete. One of the most famous combinatorial optimization (CO) problems is the Traveling Salesman Problem (TSP): it consists in finding the closed tour with minimal length that a salesman has to travel to visit exactly once each city of a given set and to return to the starting point [95, 105]. In other words, the solution of the TSP is the shortest *Hamiltonian Cycle* that connects a group of cities [163].

Variations of the basic version of the TSP include asymmetric TSPs (ATSPs) [24], pickup-and-delivery TSPs (PDTSPs) [80], multiple TSPs (MTSPs) [25], and multi-objective TSPs (MOTSPs) [30].

TSP instances are usually categorized according to the number of cities to be visited: in this respect, the literature differentiates small-scale TSPs (when the amount of cities is lower than 20), medium-scale TSPs (if the amount of cities falls in the range $[20, 100]$), and large-scale TSPs (if the amount of cities is higher than 100).

TSPs are intensively studied in computer science, numerical analysis, operations research, and engineering. They are also involved in a wide spectrum of practical problems, such as vehicle routing, cellular manufacturing, frequency assignment, circuit wiring, sensor placement, job-shop scheduling, electronic circuit design, logistics and distribution optimization, intelligent transportation, X-ray crystallography, data analysis in psychology, and wallpaper cutting, see for instance [9, 95, 98, 123, 133, 136, 138, 144] and references therein.

A full permutation of the set of the cities of interest is needed to find the solution/s of any TSP, that can be indeed considered a NP-complete problem, as its computational complexity exponentially increases with the number of locations [81, 88].

Exact methods, such as those based on graph algorithms, linear and dynamic programming, branch and bound, shortest spanning tree, and 2-matching techniques, may be therefore not able to achieve the solution/s of a given TSP instance in a reasonable computational time (and cost) [28, 79, 94].

Heuristic and meta-heuristic algorithms may be preferable in this respect: they in fact do not insist on finding the exact solution/s; rather, their goal is to obtain satisfactory (even sub-optimal) solution/s within an acceptable amount of computation and time [94, 153, 154]. Furthermore, they do not require too restrictive mathematical characteristics of the problem and have significant robustness and adaptability.

For the sake of completeness, we recall that these methods typically start from a feasible solution of a given TSP, that is then updated and improved iteratively. Such a process eventually ends when certain arbitrary conditions occur.

Heuristics and meta-heuristics methods to solve TSPs. Bio-inspired swarm intelligence (SI)-based methods are one of most popular families of heuristics and meta-heuristics techniques developed in the last few decades to solve TSP instances [175]. These algorithms employ a population of artificial agents that act, interact, and move mimicking natural or living organisms to explore the network of interest (in order to find the optimal tour). In a wide spectrum of cases, the group of probing individuals is differentiated in subfunctional units, each with a distinct behavior and role in the optimization process.

For instance, the Spider Monkey Optimization (SMO) takes inspiration from the fission-fusion behavior of this family of animals during their food searching. Its discrete version (DSMO) is applied to solve TSPs [2, 149]: in particular, the artificial group of monkeys is able to converge to short enough paths by exchanging experiences and knowledge. In this respect, time-varying subgroups, as well as local and global leaders, are identified.

The Lion Swarm Optimization (LSO) is instead based on the hunting strategy of these felines and applied, in its discrete variant, to TSP instances [33]. In this method, the animals are classified in three categories (i.e., lion king, lioness, and lion cubs) which explore the network of interest in parallel. A ring topology is then used to transfer relevant information within the population. A complete 2-Opt[1] local search is finally incorporated to improve individual solutions.

Algorithms based on Artificial Bee Colonies (ABC) are typically employed in the field of continuous optimization, see [99, 100]. However, a Swap Sequence-based version of these methods (SSABC) is applied to solve TSP instances [91]: it distinguishes three functional types of insects (i.e., employed, onlooker, and scout bees), considers a group of eight behavioral rules, and incorporates different swap operations for interindividual information exchanges. A 3-Opt algorithm, in combination with a mechanism to jump out from local optima, is included as well. The quality of TSP solutions obtained by ABC methods is shown to increase by the use of proper neighborhood selection mechanisms, as demonstrated in [152].

Other SI-based algorithms applied to TSPs include those inspired to the collective behavior of cuckoos [126], schools of fish [1], whales [26, 115, 177], cats [20], crows [61], frogs [177], gray wolfs [71, 129], bats [143], and ants (as reviewed in more details in the forthcoming part of the Introduction).

[1]Local search strategies are widely used to improve the performance of heuristic algorithms applied to TSP instances [139]. Generally speaking, these methods repeatedly try to improve the current solutions by local changes. In this respect, the k-Opt technique amounts in replacing k edges/connections of a solution with other k edges/connections that allow to reduce the tour overall length [31]. The 2-Opt and 3-Opt are particular cases, i.e., where 2, rsp. 3, edges in a solution are replaced with other 2, rsp. 3, new edges [32, 104]. 2-Opt and 3-Opt methods require the examination of O(N^2) or O(N^3) possible exchanges of arcs, being N the number of nodes in the graph of interest. This is clearly infeasible in a reasonable computation time when the number of cities is substantially large. In this respect, speed-up techniques allow to achieve run times with a subquadratic growth, see [11, 86, 139]. Strategies involving the exchange of more than 3 arcs are not commonly used, since the high computational cost is not balanced by a substantial increment in the solution quality. It is also useful to remark that the 2-Opt strategy is not directly applicable to asymmetric TSPs, because a part of the tour has to be traversed in the opposite direction. Very good solutions of a number of symmetric TSPs can found by implementing the Lin-Kernighan (LK) local search strategy [105], which considers the possibility of a time-variable number of edge exchanges. This aspect however requires a fine tuning, as commented in [86, 139]

TSP instances are also solved by the use of the Particle Swarm Optimization (PSO), as in the cases of the Velocity Tentative PSO (VTPSO) [2] and of the discrete comprehensive learning PSO [179], i.e., where good solutions are accepted according on a Metropolis criterion.

Hybridization between different SI approaches is proposed in the literature as well: for instance, in [87], the authors develop a earthworm-based deer hunting optimization algorithm (DHOA), whereas in [93], a rider optimization algorithm (ROA) is interfaced and integrated with a spotted hyena optimizer algorithm (SHOA). In both cases, the resulting techniques show good performance and sufficiently low computational complexity.

High quality solutions of TSPs are then obtained by a discretized version of the Symbiotic Organisms Search (DSOS) algorithm [53], i.e., a population-based meta-heuristic inspired by the symbiotic interactions among different organisms in nature. This approach initially creates random organisms (which represent feasible solutions of the problem) in the ecosystem of interest: subsequent adaptation occurs through different strategies and phases (including mutualism, commensalism, and parasitism), that are set to increase individual fitness and chance of survival. The DSOS algorithm is improved in [168], where excellent coefficients are used to enhance the local optimization capability, while a self-escape strategy is set to avoid stagnation into local optima. In [128], a multi-objective SOS algorithm is proposed: it is based on the definition of an adaptive penalty function suited to resolve multi-objective constrained optimization problems.

Selected TSP instances are also tackled by approaches that take inspiration from the water cycle [125] and from the laws underlying chemical reactions [107].

Genetic algorithms (GAs) are applied in this research field as well, as in the cases of a multi-offspring variant (MO-GA) [164], and of versions of these methods characterized by variable neighborhood search [40] and by several transfer operators [69]. High quality TSP solutions are obtained in [165] by a GA improved with two local optimization strategies: the former adjusts subtours composed of three edges, the latter is set to create shorter routes by reversing paths between a selected number of cities.

Good TSP solutions can be found by hybridization between GAs and other techniques. For instance, in [6], a genetic algorithm is combined both with a multiagent reinforcement learning (MARL) heuristic and with a nearest insertion into the convex hull (NICH) local search. In particular, primary tours are constructed by the MARL approach and then improved by the GA, upon the use of crossover and mutation operators, and by the NICH technique (in conjunction with a 2-Opt strategy). In [4], the authors instead employ the cross-mutation of a GA and the position update of a PSO, whereas in [71], the speed calculation at the basis of a PSO relies on a preliminary evaluation of a number of tentative tours proposed by a GA. Such a latter strategy significantly improves the accuracy of the algorithm; however, it results in a substantial increment of the calculation time.

In last years, in-depth research and development of machine learning and graph neural networks affect TSP solution strategies. In this respect, a neural combinatorial optimization framework is shown to be able to solve TSP instances [37]: in particular, the city coordinates are used as input while the neural network is trained by using reinforcement learning to predict possible optimal tours. In [84], symmetric TSPs are instead tackled by a bidirectional graph neural network, which uses imitation learning to sequentially generate the next city to visit. Generally speaking, graph neural networks are successfully demonstrated to learn the decision variables

critical for TSP solution with little supervision [135].

For the sake of completeness, we recall that TSP instances are solved also with Differential Evolution (DE) methods, even improved by the use of k-means algorithms to classify the nodes of the graph of interest [5], and with Simulation Annealing (SA) approaches, which possibly include multiple probabilities and are combined with greedy search strategies [63].

Ant Colony Optimization and its application to TSPs. As previously anticipated, a SI-based family of meta-heuristic algorithms employed to solve TSPs (and, more generally, CO problems) is the Ant Colony Optimization (ACO), that takes inspiration from the foraging behavior of these insects. Despite an almost complete blindness, ants can in fact manage to establish optimal route paths between their lairs and possible feed locations. Such an ability emerges from the following social dynamics and communication [35, 36, 68]:

- when searching for food, each ant of a given colony starts to individually explore the surrounding environment in a random manner;
- while moving, it releases on the ground a pheromone trail, whose amount depends on the quality and the quantity of possibly discovered food sources;
- pheromone presence (and distribution) is then detected by the other insects of the population, which choose (in probability) to follow the paths marked by higher concentrations of the chemical substance. An *autocatalytic* process indeed initiates [41]: the probability with which an ant chooses a route in fact increases with the number of ants that have previously chosen the same route [41];
- such a sort of positive feedback loop (i.e., defined as a process that reinforces itself) eventually results in the fact that the entire colony is guided towards the best food sources via the shortest paths, as commented in [34] and proved, in the case of a simple idealized setting, in [14]. The pheromone-dependent communication between ants is called *stigmergy*.

The algorithmic rules at the basis of ACO methods are directly taken from the above-described dynamics. In more details, to solve a TSP, a population of artificial ants is placed on the graph of interest. The insects, defined as interacting pointwise agents, are then allowed to simultaneously move along the edges of the network to complete series of tours, whose lengths are eventually calculated and compared. In particular, each individual constructs a feasible tour (which is a candidate solution of the TSP) according to the following steps:

1. it starts from a randomly chosen node/city;
2. it moves from its current location to one of the unvisited nodes according to a set of rules that varies in the different ACO instances. The traversed edge is then added to the solution under construction;
3. in the absence of unvisited nodes, the ant closes its present tour by moving back to the starting city, being therefore ready to construct a new solution of the TSP.

In accordance to the ecology, moving agents are assumed to release a given amount of pheromone on the edge/s they actually traverse. In particular, updates of the chemical pattern, that possibly include also evaporation aspects, may occur either at each construction step (i.e., after each single move of the ants from a node to another one) or at the end of an entire tour of the graph. The overall algorithm is finally iterated until a given stopping condition is satisfied.

The naive versions of method are proposed by Dorigo and colleagues in the early 1990's and commonly denoted as Ant System (AS, see [41, 42, 44, 45, 46]). A first AS variant is characterized by the fact that, at each step of the algorithm, the artificial insects choose the next destination node, among those permitted, according to a probabilistic rule which depends *only* on the spatial pattern of pheromone, i.e., edges of the graph characterized by higher chemical amounts are more desirable. In an improved AS version, the decision of an ant on the next destination is instead set to depend not only on pheromone distribution but also on the distance between its actual location and the remaining unvisited nodes. In this case, the chemical trail can be updated either at each construction step (giving rise to the Ant-Density and the Ant-Quantity variants of the AS) or at the end of any given ant tour (giving rise to the Ant-Cycle variant of the AS). The edge length (or the inverse of this quantity, called *visibility*) can be interpreted as a heuristic information associated to each feasible solution component [14].

Several extensions of the AS are then introduced over years. For instance, the Ant-Q algorithm [59] integrates in the method some ideas from the Q-learning [170]: in particular, it uses a pseudorandom state choice rule. In [41, 42, 44], the same group of authors introduces an *elitist strategy*: it amounts in the fact that, after every completed ant tour, the pheromone on the edges belonging to the actual best solution of the problem is reinforced by a further quantity, which is proportional to the number of elite ants and inversely proportional to the length of the minimal tour itself. An appropriate amount of elitist agents is indeed shown to result in better tours obtained in a reasonably low computational time: in this respect, when their number is too large, the search early concentrates around suboptimal solutions, i.e., an early *stagnation* emerges.

In the so-called rank-based ACO algorithm (AS-rank, see [21]) selected quantities of chemical are added on the edges belonging not only to the actual optimal solution (as in the case of the elitist strategy) but also to a given number of suboptimal ones. In particular, the amount of the chemical extra-depositions is inversely proportional to the length of the corresponding tours. In the same work, it is shown that, for a large TSP instance (i.e., based on a graph with 132 nodes), the AS-rank and the elitist AS perform significantly better than the naive AS (being superior also to genetic algorithms and simulated annealing procedures), with the former approach giving slightly better results than the latter one.

It is useful to underline that, as observed in [14], the pheromone further additions accounted in the above-described approaches are established by a sort of "daemon" action, as they require the knowledge of global information.

A number of strategies, that mostly affect the pheromone update rule, are then developed to avoid stagnation, i.e., the situation in which all ants follow the same suboptimal path. In this respect, the Ant Colony System (ACS, see [46]), which is at the basis on many subsequent ACO variants, is characterized by two modifications: (i) the pheromone is added only to the edges belonging to the current best solution whereas it is partially removed from the other arcs (even they have been traversed by an ant) and (ii) at any iteration of the algorithm, each artificial insect

has two possible choices, both characterized by a given probability, i.e., it either deterministically travels the edge with the highest pheromone level or performs a biased exploration of the remaining permitted edges. The former strategy decreases the attractively of the edges belonging to suboptimal solutions and favors the exploration of unvisited arcs, whereas the latter one represents a compromise between the pseudorandom state choice rule typically used in Q-learning and the random-proportional action one employed in the naive AS.

In the $\mathcal{MAX} - \mathcal{MIN}$ Ant System ($\mathcal{MMAS}$ [156, 160]), the pheromone amount on any arc of the graph is constrained to fall within a given interval, so that the selection probability of any edge can not drop below a certain threshold. Furthermore, the distribution of the chemical is initialized to the upper limit along the entire graph, whereas the substance is added only to the arcs belonging to the current best tour (like in ACS approach). These strategies, if an appropriate choice for the trail limits is done, reduce the possibility of stagnation around suboptimal solutions, as they enhance the exploratory capacity of the algorithm [155].

In [97], the $\mathcal{MMAS}$ approach is enriched by an elitist strategy to further increase the convergence speed and the solution quality.

Modifications of pheromone kinetics, in order to obtain shorter tours of a given topology of cities, are proposed also in [108] and in [173]. In particular, in the latter case, the authors introduce variations in the chemical evaporation rule, which allow to achieve a better balance between solution quality and convergence speed.

In [101], the chemical is added on the edges belonging to current best solution, whereas it is significantly reduced on the arcs composing the current worst path. Such a strategy, which can be interpreted as sort of combined positive/negative feedback mechanism, is shown to improve the convergence speed of the algorithm towards optimal tours of the graph.

The performance of ACO methods in solving TSPs can be improved by differentiation of the ant population. In this respect, in [52], the authors distinguish different subcolonies, which are set to perform different exploratory actions, and propose a transformation strategy. Similarly, in [70], a parallel cooperative ACO (PACO) is proposed: it splits the ants into multiple groups, that independently explore the graph (employing a 3-Opt strategy) and share information. The overall best solution is then generated by comparing the shortest tour found by each subcolony. Diversity within the colony of artificial insects is obtained in [161] by the use of a transfer strategy, based on a large neighborhood search: this approach is shown to have the ability to quickly obtain very good solutions in a reasonably low computational time. Heterogeneity in the behavior of the population of ants is instead obtained in [48] by classifying the cities of the graph with a k-means algorithm and by setting a specific movement rule for any subgroup of nodes. The resulting method (denoted as AHACO) is further integrated with a modified 2-Opt local optimizer and with a tailored mechanism to jump out from local optima (i.e., to avoid stagnation).

A local search operator, that improves TSP solutions by removing and inserting cities, is integrated in a Memetic ACO algorithm to solve dynamic traveling salesman problem (DTSP), see [116]. For the sake of completeness, we remark that the same authors include an Immigrants Scheme in a similar approach [117].

Good performances of ACO methods can obtained by reducing, for any ant and at any given construction step, the set of possible target cities by applying some restrictions based on the

heuristic information represented by the internodal distance [57].

In most of ACO approaches applied to TSPs, the parameter setting is initialized and remains constant for the entire search process. In this respect, if the coefficients of the algorithm are not properly fixed, the solution quality is poor and the possibility of stagnation increases, as commented in [141]. To address this issue, a dynamical parameter tuning is proposed by an increasing number of authors.

For instance, in [49], a dynamic pheromone evaporation (DPE) method is adopted within an ACO framework. The resulting algorithm (denoted as DEACO) is also characterized by the fact that the nodes of the graph are classified using a clustering method, whose output is used to intelligently select the starting city of a tour. The integration of the above-described strategies is shown to reinforce the exploration of the entire search space, thereby preventing premature convergence of the algorithm to suboptimal solutions, and to give a good performance also in terms of evolution speed.

In [7], the values of selected parameters characterizing an ACS algorithm are adjusted during the search process according to the following strategy. The ant population is differentiated in several subgroups: each of them is assigned a distinct set of parameters and independently explores the graph (with a local optimization approach). The solutions found by the subcolonies are then evaluated and compared and the best one is used to have an indication of how the parameter setting has to be updated.

Similarly, in [118], the authors assign to each of the several subcolonies in which the ant population is differentiated a distinct set of parameter values: at the end of any tour, the group of insects that has obtained the current worst solution is assigned (via a mutation operator) a small modification of the parameter setting characterizing the group of agents that has instead found the current best solution.

Estimates of ACO coefficients are obtained by an orthogonal experiment method in [60] and by a Taguchi approach in [131]. In both cases, numerical realizations demonstrate that a reasonable parameter configuration can be obtained with a substantially low number of experiments.

Modification of the heuristic information is instead at the basis of the algorithm described in [148]. In particular, the authors propose an ACO-based method with Adaptive Visibility (ACOAV), which adopts a transition rule emphasizing not only the proximity between the actual ant position and possible target destinations but also its distance to the returning (i.e., starting) city. A 3-Opt local search operation is also incorporated.

In [162], a Heterogeneous Adaptive ACO method (HAACO) to tackle TSPs is initialized with a population of ants sharing the same greedy behavior. Differentiation within the colony is then introduced by a set of rules that allows to vary, during the search process and independently for each insect, the parameters controlling the trade-off between the movement stimuli deriving form either the heuristic information or the pheromone trail. In particular, the HAACO algorithm is able to retain the fittest individual/s in the population through a sort of elitism mechanism: the best-performing ant is in fact set to only undergo little mutation, whereas the offspring replaces the worst-performing one. In particular, uniform and Gaussian mutation operators are investigated, with better results obtained by the latter one. Through the use of selection, elitism and mutation, the proposed strategy (which also include a 3-Opt algorithm) is indeed able to generate and conserve promising parameter settings. The iterative cycle of exploration and exploitation

of the parameter space reflects the dynamic nature of search processes.

A recent trend involves hybridization between ACO algorithms and other SI-based methods, in particular in the perspective of parameter optimization. For instance, in [67, 76, 113], tailored versions of the PSO are used to iteratively adjust selected coefficients controlling ant movement, such as the weights of the heuristic and of the chemical stimuli as well as the pheromone evaporation rate. In the former case, the number of ants is varied as well, whereas, in the latter case, after the termination of ACO/PSO information exchanges, the already-cited 3-Opt algorithm is employed to jump out from local optima, thereby boosting the accuracy of the solution and the robustness of the method. Coordination between PSO, ACO, and k-Opt techniques is also explored in [92]. Similarly, in [50], the PSO uses the pheromone pattern obtained by an ACO-based algorithm to bias the movement of its particles along a given graph.

In [176], a ACS is integrated with the Harris's Hawk Optimization (HHO) method, which is commonly employed to solve a wide spectrum of problems, see [39, 77, 142]. In particular, the HHO is used, as the PSO in the previously-cited cases, to reasonably adjust ACS parameters. Hybridization between selected ACO methods and Dragonfly and Firefly Algorithms (rsp., DA and FA) is proposed in the literature as well [8, 120].

Optimization of ACO parameters can be also obtained by integration with other families of heuristics and meta-heuristics. For example, in [51], the authors run the ACO algorithm several times, in each case with a distinct couple of coefficients regulating the pheromone and the heuristic stimuli: the obtained solutions are then used by an artificial neural network approach to predict optimal parameter settings.

Good-performing ACO coefficients can be derived by running GAs, see [19, 56, 96, 134] and references therein. In particular, in most of these approaches, the ant system is initialized by a random combination of parameters, that are then properly adjusted over time by the employed GA. The solution of a given TSP obtained with a GA is instead used in [178] to infer the initial pheromone pattern of an ACO algorithm, whose global search ability and speed of convergence are significantly improved.

Similarly, in [54], a graph convolutional network approach is employed to generate a reasonable solution of a TSP instance, which is converted into a pheromone distribution and passed as initial condition to a ACO-based method. The resulting approach (denoted with the acronym GCNIACO), which represents a relevant example of combination between machine learning and swarm intelligence, is then enriched by a dynamically-adjusted chemical volatility factor and by a 3-Opt strategy to jump out from local optima.

It is useful to underline that the above-mentioned algorithms used to optimize ACO parameters, such as those belonging to the classes of PSO and GA, have their own set of characteristic coefficients, which indeed affect the performance of the entire hybrid computational framework. An advantage in this respect is represented by the combination of ACO variants with the previously-cited SOS method, whose performance is not substantially influenced by the specific parameter setting, as commented in [27]. This strategy is employed in [169], where a ACO approach uses coefficient values preliminarily optimized by a SOS. Also in this case, a local optimization strategy is incorporated into the hybrid algorithm to increase the convergence rate and the solution quality.

Ant Colony Optimization and its application to selected practical CO problems. For the sake of completeness, we now provide a brief overview of the application of selected ACO-based methods to practical CO problems.

In [137], the previously-cited Taguchi experimental design is employed in combination with an ACO algorithm to improve load leveling in cloud computing environments. In particular, the proposed approach uses a fuzzy module to assess optimal pheromone value, in order to speed the computation and to avoid stagnation.

In [102], the authors apply to a planning problem of unmanned cranes during hoisting a ACO variants characterized by variations in the heuristic function, in the pheromone update mechanism, and in the path selection formula.

The Improved Dynamic Adaptive Ant Colony Optimization (IDAACO), described in [109], is instead based on (i) the inclusion of directional information in the heuristic function, (ii) the use of an adaptive pseudo-random transmission strategy, and (iii) a modification of both local and global pheromone update mechanisms: it is then shown to have excellent performance to handle the problem of pipe routing design.

Path planning problem of mobile robots are instead efficiently tackled in [174] by a ACO version that includes the construction of a multi-factor heuristic function and a stepwise initial distribution of pheromone, whose kinetics are (i) subjected to the above-described $\mathcal{MAX} - \mathcal{MIN}$ concentration constraint and (ii) characterized by an adaptively-adjusted volatility factor.

A time-varying chemical evaporation rate is present also in [78], where the authors address a vehicle routing problem with an Improved Ant Colony Optimization (IACO) algorithm, whose main features are a modified transition probability law and the introduction of a variable neighborhood local search, based on the use of insertion and exchange operator. Such a last aspect, coupled by a pheromone update mechanism relying on information entropy, is at the basis of the ACO method proposed in [167] and used to address a similar problem.

Motivation of the work. As it is possible to conclude from the above literature overview, ACO-based methods have been applied with considerable success to a wide variety of combinatorial problems stemming from both academic and real-world applications. Generally speaking, this family of algorithms is particularly suited to deal with discrete optimization problems, since it employs a strategy of a step-by-step solution construction. In particular, in the case of TSPs, ACO approaches have shown strong robustness and a significant capacity to discover solutions with a reasonably high quality (especially in the case of inclusion of local search methods).

On the other hand, the TSP has played (and still plays) an important role in the development of ACO-based methods, since the first and naive version of the algorithm, i.e., the Ant System, as well as many of the subsequently proposed variants, have used such a specific combinatorial problem as a benchmark test to assess their performance. As commented in the comprehensive book by Dorigo and Stützle [45], there are several reasons underlying this choice: i) the TSP is an important NP-hard optimization problem, that arises in several applications, and it is widely recognized as a standard test bed for new algorithmic ideas; ii) the TSP is a problem to which ACO approaches can be straightforwardly applied with easily understandable results, i.e., the algorithm behavior is not obscured by too many technicalities. Finally, as stated by the two authors

> *"... the history of ACO shows that very often the most efficient ACO algorithms for the TSP were also found to be among the most efficient ones for a wide variety of other problems ...* ([45], Pag. 65)".

In this work, we introduce the AddACO, a bio-inspired modified version of the canonical ACO algorithm. Its name originates from the fact that the probabilistic law regulating ant decision on the next destination (among those permitted) has here an additive form and not the classical multiplicative one. In other words, the exploratory behavior of the artificial insects here results from a linear convex combination of competing migratory stimuli, rather than by their superimposition. The mathematical structure given to the ant transition rule conceptually improves classical ACO methods in at least two aspects: (i) it allows to prevent a simultaneous minimization/maximization of all migratory guidance cues and (ii) it accounts for their possibly independent effects. In this respect, a given edge of the graph maintains the possibility to be explored even if a subset of migratory traits locally drops to zero.

Variants of our algorithm stem from the specific set of decisional variables assumed to affect ant movement: in this respect, the AddACO-V1 employs pheromone trails and proximity as insect behavioral stimuli, whereas the AddACO-V2 and the AddACO-V3 also include randomness and inertia, respectively.

Our approach does not include strategies to increase solution quality, e.g., methods to improve local search or parametrization. The aim is in fact not to develop an algorithm that outperforms those already present in the literature (and reviewed in the previous parts of the Introduction) but to propose an *alternative* ACO-based method, which is characterized by the above-described conceptual/theoretical improvements while maintaining a sufficient level of performance. In this respect, series of benchmark tests are used in the forthcoming sections to assess whether the proposed additive version of the ant decisional rule reduces the exploratory ability of the insect population with respect to the canonical transition law. In particular, for a fair and consistent analysis, our method is compared to the Ant-Cycle AS that is, as seen, a naive ACO approach whose performance entirely relies on the efficiency of the ant transition probability, being not affected by the use of other (and more sophisticated) exploratory strategies.

Outline of the article. The rest of the paper is organized as follows. The formalization of the TSP is given in Section 2. In Section 3, we present the description of the AddACO and of its variants, with comments on the component ingredients and parameters. Section 4 is devoted to show selected numerical results. In particular, in Subsection 4.1, the three versions of our method are applied to two middle-size graphs, in order to analyze their performance and behavior upon variations in their characteristic coefficients. In Subsection 4.2, the best-performing variant is then tested on some representative large-size TSP instances and compared to the Ant-Cycle version of the AS approach proposed by Dorigo and coworkers in [41]. Review and discussion of the numerical results, as well as hints for further improvements of the algorithm, and on its application to other optimization problems, is contained in Section 5. Analytical results on the asymptotic behavior of our version of the method are finally provided in the Appendix of the article.

2. The Travel Salesman Problem (TSP): Formal Description and Notation

Following [14, 130], any combinatorial optimization (CO) problem can be formally defined as $\mathcal{P} = (\mathcal{S}, f)$, where

- $\mathcal{S}$ identifies the finite set of feasible solutions (that is also called *search space*);
- $f : \mathcal{S} \longrightarrow \mathbb{R}^+$ is the objective function, that assigns a positive cost value to each element $s \in \mathcal{S}$.

The goal is indeed to find, among all candidate solutions of the problem, those that are characterized by the minimal cost value, i.e., those that fall within the set

$$\mathcal{S}^* = \left\{ \arg\min_{s \in \mathcal{S}} f(s) \right\} \subseteq \mathcal{S}. \tag{1}$$

In other words, a solution s^* belongs to $\mathcal{S}^*$ if and only if it minimizes the objective function f.

From a mathematical perspective, any TSP can be formalized as follows. First of all, we notice that an instance of such a type of CO problem is defined by a bidimensional graph $\mathcal{G} = (V, E)$ where, as shown in Fig. 1,

- $V = \{n_i\}_{i=1}^{N}$ is the set of nodes/vertices, each representing a town, N being their overall number;
- $E = \{e_{ij} = \{n_i, n_j\}\}_{\substack{i,j=1,\dots,N \\ i<j}}$ is the set of edges/arcs, each linking a pair of nodes. In particular, e_{ij} denotes the edge between towns n_i and n_j (i.e., the edge *incident* to n_i and n_j) whose length in the Euclidean norm is hereafter identified by $l_{e_{ij}}$. By using the same terminology as in [14], the distance between each pair of cities represents a sort of weight of the corresponding edge and can be interpreted as a heuristic information.

Typical assumptions on $\mathcal{G}$ are:

- it is *simple*, in the sense that it does not contain self-loops and parallel edges (i.e., each pair of nodes is connected by a unique arc);
- it is *undirected*, in the sense that its arcs can be traveled in both directions. In this respect, each edge has been previously defined just as set of two nodes. In the case of a directed graph, each arc should be instead defined as an *ordered* pair of nodes;
- *connected*, in the sense that every pair of nodes has an incident edge.

Any sequence of connected vertices (or of corresponding edges) defines a *path* in $\mathcal{G}$. A *cycle* is formally a path where the start node is the same as the end node, whereas all the other component vertices are distinct. A cycle that contains all vertices of the graph is finally defined as a *tour*, see Fig. 1 (B-C). As sketched in the Introduction section, a tour can be also regarded to as an *Hamiltonian* path $H = \{V, E\}$. In this respect, the overall set of possible tours of a given graph $\mathcal{G}$ can be identified by $\mathcal{C}(\mathcal{G}) = \{c_k\}_{k=1}^{K=N!}$, N being the cardinality of V, i.e., the number of nodes

of $\mathcal{G}$. The length of the generic k-th tour, denoted by l_{c_k} is finally defined as the sum of the extensions of the component edges. Two different tours are hereafter considered *equivalent* if they have the same length, e.g., $c_{k_1} \sim c_{k_2} \iff l_{c_{k_1}} = l_{c_{k_2}}$ being $k_1 \neq k_2$, see panel (C) of Fig. 1. For instance, we consider equivalent:

- tours composed by the same sequence of arcs but traveled in opposite directions;
- tours composed by the same sequence of edges but characterized by different starting/ending nodes.

By using the notation introduced above for a generic CO problem, a TSP instance can be indeed formalized as follows. Given the graph $\mathcal{G}$ of interest, with the above-defined properties:

- the search space amounts in the set of feasible tours, i.e., $\mathcal{S} = \mathcal{C}(\mathcal{G})$;
- the objective function f associates to each tour its overall length, i.e., $f(c \in \mathcal{C}(\mathcal{G})) = l_c$.

The solution of the problem is indeed given by the subset $\mathcal{C}^*(\mathcal{G}) \in \mathcal{C}(\mathcal{G})$ such that:

$$\mathcal{C}^*(\mathcal{G}) = \left\{ \underset{c \in \mathcal{C}(\mathcal{G})}{\arg\min} f(c) = l_c \right\} \subseteq \mathcal{C}(\mathcal{G}). \tag{2}$$

In this respect, as observed in [14], the arcs of $\mathcal{G}$ can be referred to as possible *solution components*, since they are the units from which candidate solutions are assembled.

Remark. We used (and will use) the letter "c" to indicate concepts and quantities relative to the tours of a given graph (e.g., $C(\mathcal{G})$, l_{c_k}, ...), since the letter "t" will be used for the time-variable.

3. Proposed Algorithm: the AddACO

We here propose an ACO-based algorithm characterized by a substantial modification in the mathematical structure of the transition probability affecting ant decision on the next destination.

In this respect, let us first define with t the discretized time variabile (i.e., $t \in \mathbb{N}$, $t \geq 0$) and with M the (constant) number of insects disposed on the graph of interest $\mathcal{G}$, which have an initial arbitrary (user-defined) spatial distribution. At the onset of the algorithm, each artificial insect is indeed allowed to move towards all nodes except from the starting one.

We then set that, for any time step $t > 0$, the m-th ant (being $m = 1, ..., M$), actually located in the i-th town, moves along the graph according to a *transition probability* of this type:

$$p_{n_i \to n_j}(t) = \begin{cases} \displaystyle\sum_{b \in \mathcal{B}} \alpha^b \frac{\phi^b_{e_{ij}}(t)}{\displaystyle\sum_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} \phi^b_{e_{ik}}(t)}, & \text{if } n_j \in V^m(t); \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

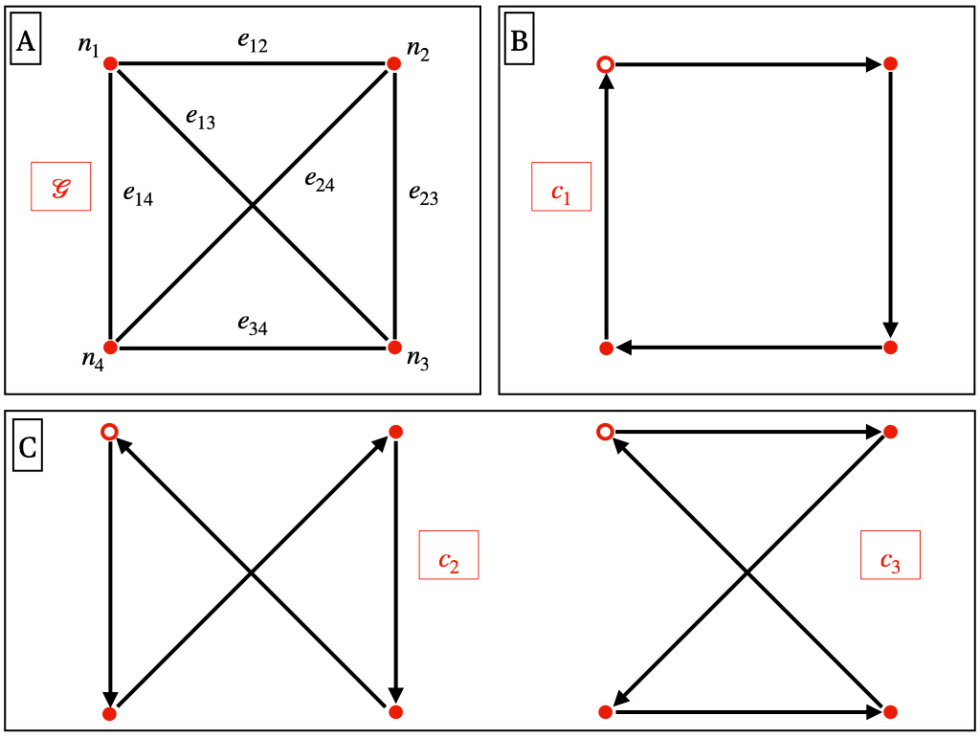

Figure 1: (A) Representative graph $\mathcal{G}$, composed of four nodes connected by 6 edges, i.e., $\mathcal{G} = (V = \{n_1; n_2; n_3; n_4\}, E = \{e_{12}; e_{23}; e_{34}; e_{13}; e_{14}; e_{24}\})$. (B-C) Three possible tours of $\mathcal{G}$, all starting from node n_1. The tour $c_1 = (n_1, n_2, n_3, n_4, n_1) \equiv (e_{12}; e_{23}; e_{34}, e_{14})$ is the minimal length one, whereas the tours $c_2 = (n_1, n_4, n_2, n_3, n_1) \equiv (e_{14}; e_{24}; e_{23}, e_{13})$ and $c_3 = (n_1, n_2, n_4, n_3, n_1) \equiv (e_{12}; e_{24}; e_{34}, e_{13})$ are equivalent, according to the definition given in the text. In fact, assuming $l_{e_{12}} = l_{e_{23}} = l_{e_{34}} = l_{e_{14}} < l_{e_{13}} = l_{e_{24}}$, we have that $l_{c_1} < l_{c_2} = l_{c_3}$. In this respect, $c_1 \in \mathcal{C}^*(\mathcal{G})$, cf. Eq. (2).

where $V^m(t) \subset V$ is the set of permitted target nodes for the m-th insect at time t, i.e., those not already visited during its current tour. $\mathcal{B}$ instead indicates the set of behavioral traits/external stimuli that are assumed to affect ant migratory decision: each of them is quantified for any arc of the graph (i.e., for any feasible solution component) by the variable $\phi_e^b(t) \in \mathbb{R}_+ \cup \{0\}$, being $e \in E$.

The generic coefficient α^b is indeed a *weight* that establishes the relative importance of the corresponding b-th stimulus w.r.t. the others. In particular, to have comparable effects and to avoid a simultaneous minimization/maximization of all migratory inputs, we set that:

$$\begin{cases} \alpha^b \in [0, 1], \quad \text{for any} \quad b \in \mathcal{B}; \\ \sum_{b \in \mathcal{B}} \alpha^b = 1. \end{cases} \tag{4}$$

We are indeed assuming that ant exploratory dynamics result from a linear convex combination of *competing* stimuli, rather than from the superimposition of *cooperating* behavioral traits, as it is classically set in ACO-based algorithms. According to us, the mathematical structure of Eq. (3) has a clear *biological/ecological inspiration* and represents a conceptual improvement for

this family of meta-heuristics in at least two aspects:

- artificial agents are prohibited to simultaneously minimize or maximize *all* included movement stimuli;
- each migratory trait acts independently from the others, i.e., its magnitude does not have an impact on the others. This model modification allows to avoid the situation in which a given edge of the graph loses the possibility to be travelled even in the case of *a unique* ant decision variable locally (i.e., on that arc) equal to zero, as it would instead result with the canonical multiplicative form given to the transition probability in most ACO approaches.

In the following sections, we will assess whether the proposed variations in the migratory rule of the population of artificial insects have the side-effect to reduce the exploratory potential of the algorithm.

For the sake of terminological simplicity, we hereafter denote with the acronym *AddACO* our version of the method.

Starting from $t = 0$, the ants use blocks of N iterations to complete their tours, as $(N - 1)$ moves are necessary to visit the entire group of towns and a further one is needed to come back to the starting node. Every zN time steps ($z \in \mathbb{N}$, $z > 0$), the sets of permitted destinations of all ants (i.e., V^m for $m = 1, .., M$) are then reinitialized and the lengths of the just-terminated tours are computed and compared to the minimal one found so-far, which is eventually updated.

The overall algorithm is finally iterated for $K_{\text{MAX}}N$ time steps, where $K_{\text{MAX}} \in \mathbb{N}$ ($K_{\text{MAX}} > 0$) identifies an arbitrary number of tours that the ant population is allowed to perform in sequence. However, the numerical procedure is set to stop also when all insects *simultaneously* perform *equivalent* tours (i.e., tours composed by the same sequence of edges, see the definition given in the previous Section 2), even starting from different positions: such a situation is hereafter called *uni-path* behavior and results in the impossibility for the algorithm to search for alternative solutions, see also the comment by Dorigo and colleagues in [41].

Given the above-general structure of the proposed algorithm, three specific variants are now described, each deriving by a distinct set of ant migratory behavioral traits.

AddACO-V1. The first version (V1) of our method is similar to classical ACO approaches. The effective decision of an ant on its destination node is in fact assumed to be a trade-off between its tendency to go towards closed enough towns (i.e., to be affected by the heuristic information) and the desire to follow highly trafficked edges, i.e, as quantified by the amount of pheromone laid by its groupmates. In this respect, $\mathcal{B} = \{\text{pheromone trail; node proximity}\}$, so that Eq. (3) rewrites, in the case of the representative m-th agent and pair of nodes n_i and n_j, as

$$p_{n_i \to n_j}(t) = \begin{cases} \alpha \dfrac{q_{e_{ij}}(t)}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} q_{e_{ik}}(t)} + \beta \dfrac{(l_{e_{ij}})^{-1}}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} (l_{e_{ik}})^{-1}}, & \text{if } n_j \in V^m(t); \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

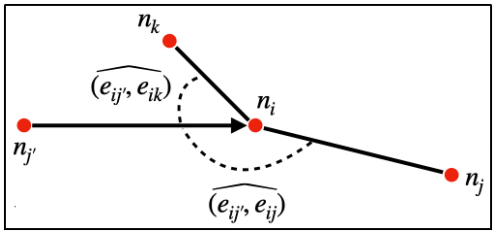

Figure 2: With the inclusion of a persistence contribution in the transition probability, the ants have to evaluate the alignment of the possible next movement to their previous one, see Eq. (7). In this respect, the target destination n_j is preferable to the target destination n_k for an artificial agent that is actually located in the node n_i and comes from the node $n_{j'}$, as $(\widehat{e_{ij'}, e_{ij}}) > (\widehat{e_{ij'}, e_{ik}})$.

where $q_e(t)$ is the quantity of chemical substance present at time t on the edge $e \in E$, whose length is denoted by l_e (see Section 2). For the sake of completeness, we recall that the quantity $\eta_e = l_e^{-1}$ gives the so-called *visibility* of the generic arc $e \in E$, as defined in the work by Dorigo and colleagues [41]. According to (4), in Eq. (5), we have that $\alpha, \beta \in [0, 1]$ and $\alpha + \beta = 1$.

AddACO-V2. In the second version (V2) of our method, we include a noise contribution in ant exploratory dynamics. In this respect, we set $\mathcal{B} = \{\text{pheromone trail; node proximity; noise}\}$, while the decision probability rule (3) is specified as:

$$p_{n_i \to n_j}(t) = \begin{cases} \alpha \dfrac{q_{e_{ij}}(t)}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} q_{e_{ik}}(t)} + \beta \dfrac{(l_{e_{ij}})^{-1}}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} (l_{e_{ik}})^{-1}} + \gamma_{rand} \dfrac{\mathcal{U}_{e_{ij}}(t)}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} \mathcal{U}_{e_{ik}}(t)}, & \text{if } n_j \in V^m(t); \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

in the case of the representative m-th agent and pair of nodes n_i and n_j. In the above Eq. (6), the first two terms are the same as in (5), whereas $\mathcal{U}_e(t)$ is a random function uniformly distributed over the interval $[0, 1]$, calculated at each construction step t for any edge $e \in E$ of the graph. It biases ant decision on the next destination. Consistently with the previous considerations, we assume that $\alpha + \beta + \gamma_{rand} = 1$, with $\alpha, \beta, \gamma_{rand} \in [0, 1]$.

AddACO-V3. We finally introduce an inertial aspect in the ant decision to move towards a given node of the graph, i.e., we consider the fact that individuals (not only animals but also cells and bacteria) prefer to move along their actual path without undergoing sudden directional variations. From an algorithmic point of view, we indeed set $\mathcal{B} = \{\text{pheromone trail; node proximity; inertia}\}$ while, for a given insect m and pair of nodes n_i and n_j, the transition probability (3) is modified as:

$$p_{n_i \to n_j}(t) = \begin{cases} \alpha \dfrac{q_{e_{ij}}(t)}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} q_{e_{ik}}(t)} + \beta \dfrac{(l_{e_{ij}})^{-1}}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} (l_{e_{ik}})^{-1}} + \gamma_{inert} \dfrac{\widehat{(e_{ij'}, e_{ij})}}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} \widehat{(e_{ij'}, e_{ik})}}, & \text{if } n_j \in V^m(t); \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

where $\widehat{(e_{ij'}, e)}$ indicates the convex angle between a generic arc $e \in E$ and the edge $e_{ij'}$, being $n_{j'}$ the node where the m-agent was previously located, i.e., at the discretized time $t-1$. The third term in (7) indeed states that the ant m more probably chooses to travel an arc that is somehow aligned with its past movement, see Fig. 2. The first two contributions in Eq. (7) are those defined in (5), and also included in (6). In accordance with (4), $\alpha + \beta + \gamma_{inert} = 1$, with $\alpha, \beta, \gamma_{inert} \in [0, 1]$.

Pheromone kinetics. All above-defined variants of our ACO-based algorithm include the effect of the pheromone trail, whose spatio-temporal update rule is assumed to be the same in the three cases. In more details, variations of the chemical pattern are set to occur at every zN iterations ($z \in \mathbb{N}$, $z > 0$), i.e., after each complete ant tour. For any representative edge $e \in E$, we indeed write:

$$q_e(t = zN) = q_e(t = (z-1)N) + \sum_{m=1}^{M} r(l_{c_z^m}) \mathbf{1}_{m \in S_e^z}, \tag{8}$$

where $\mathbf{1}$ is the indicator function and S_e^z defines the set of ants that have included the arc e during their last tour. In this respect, c_z^m identifies the tour completed in the z-th block of iterations by the m-th insect, whose corresponding length is equal to $l_{c_z^m}$. Such a quantity in turn affects the amount of pheromone released by the m-th moving agent, as

$$r(l_{c_z^m}) = \frac{r}{l_{c_z^m}}, \tag{9}$$

r being an arbitrary constant hereafter fixed equal to 1. As demonstrated by Dorigo and colleagues in [41], modifications in the value of this parameter are in fact substantially irrelevant in terms of numerical outcomes. Eq. (8) recalls the Ant-Cycle version of Dorigo's Ant System for two reasons: (i) chemical kinetics do not occur at every construction step, i.e., after each single ant move, and (ii) arcs belonging to shorter tours are made more desirable also by the trail spatial pattern [41]. Such a last aspect is widely shown to increase the algorithm search efficiency [41, 42]. We do not account for pheromone evaporation in order to limit the number of model parameters, considering the fact a homogeneous decay term (i.e., equal for any arc of the graph) would not affect the transition probabilities defined in Eqs. (5)-(7).

4. Numerical Results

In this section, we analyze the performance of the proposed variants of our ACO method in the case of selected TSP instances. An analytical study of their asymptotic characterization is instead given in the Appendix of the work.

The three versions of our algorithm are coded using a C++ language, with Visual Studio 2019 as IDE, and run on a PC with Intel(R) Core(TM) i5-5200U CPU (2.20 GHz-8 GB RAM) and the Windows 7 64-bit operating system.

In particular, for any forthcoming simulation setting, 20 independent numerical realizations are run: each of them amounts in a maximum number of ant tours equal to $K_{\text{MAX}} = 5000$, i.e., a value sufficiently high to establish the algorithmic behavior, as we observed with means of preliminary simulations and also commented in [41]. The following outputs are then recorded:

- BFT (best-found tour), which is the length of the minimal tour found among the 20 runs. In this respect, we hereafter denote with BKT (best-known tour) the length of the optimal Hamiltonian cycle of a given graph known in the literature (obtained by either exact or approximated methods). This quantity is taken as a reference value to assess the quality of the solutions of our approach;

- $V = \dfrac{AFT - BFT}{AFT} * 100$, where BFT is the best-found tour whereas ***AFT*** (average-found tour) is the mean value of the length of the minimal tours found in the 20 runs. V evaluates (in percentage) the variability of the solutions obtained by an algorithm, i.e., its ability to consistently find the best possible results. The smaller the V is the more a numerical method is in fact stable and robust (the reader may refer to [169] for comments on this aspect);

- UPB (uni-path behavior), i.e., it is a dichotomous variable that is set to be equal to "yes" if the ant population converges to equivalent tours in all 20 runs (otherwise it is set equal to "no"). In the case of uni-path behavior, we also evaluate the average number of tours needed to have such a convergence.

On the basis of the values of the above-defined observables, we can distinguish the following four scenarios:

- S_1, which consists of a uni-path behavior where all ants constantly find equivalent optimal tours. It is the best situation, identified by $BFT = BKT, V = 0$, and $UPB =$ "yes";

- S_2, which is characterized by the absence of uni-path behavior but by the ability of a subset of ants to find an optimal tour in all numerical runs. This situation is identified by $BFT = BKT, V = 0$, and $UPB =$ "no";

- S_3, which consists in a uni-path behavior where all ants converge to suboptimal equivalent tours in all runs. This is the situation of *stagnation*, here identified by $BFT > BKT, V > 0$, and $UPB =$ "yes" that, as commented in the Introduction, represents one of the main issues of ACO-based algorithms applied to solve TSP instances;

- S_4, which is characterized by the absence of uni-path behavior and by the fact that no ant is able to find an optimal tour in any of the simulation runs. It is the worst situation, here identified by $BFT > BKT, V > 0$, and $UPB =$"no".

The forthcoming simulation results will be then grouped in two parts:

1. in Subsection 4.1, the three variants of our algorithm are applied to two middle-size graphs, i.e., in order to analyze their behavior upon variations in their characteristic coefficients and to compare their performances;

2. in Subsection 4.2, the best-performing variant found out in Subsection 4.1 (equipped with its optimal parameter setting) is tested on some representative large-size TSP instances and compared (in terms of search potential) to the Ant-Cycle version of the AS approach proposed by Dorigo and coworkers in [41]. The choice is justified by the fact that, as stated in the Introduction, our aim is to demonstrate that the additive form of the transition probability introduced in this work is competitive w.r.t. the canonical moltiplicative one, i.e., it represents a sufficiently good alternative since it does not have a negative impact on the exploratory ability of the insect population. In this respect, it is natural (and fair) to not compare our algorithm to more sophisticated ACO-based approaches, which possibly include a large spectrum of strategies that optimize their search process and allow them to outperform our method, as confirmed by the data in Tables 8 and 9, see the conclusive section of this work.

4.1. Parameter Analysis of the AddACO Variants

The analysis of the influence of selected parameters on the exploratory ability of the proposed ACO variants is conducted with experiments on the following graphs:

- $\mathcal{G}_{6\times6}$, i.e., a regular grid of 36 evenly spaced nodes. For this topology of cities, the optimal tour is *a priori* known: its length is equal to 36 times the dimension of the edges that vertically and/or horizontally connect two vertices of the lattice, here set to 10 non-dimensional units, see Fig. 3 (A). Accordingly, $BKT(\mathcal{G}_{6\times6}) = 360$;

- $\mathcal{G}_{30}$, which identifies the graph at the basis of the *Oliver 30* problem, a 30-cities benchmark distribution of nodes [82, 132]. For this TSP instance, the Ant-Cycle method finds the optimal tour shown in Fig. 3 (B), whose length is equal to 423.74 non-dimensional units, that is here assumed to be the value of $BKT(\mathcal{G}_{30})$. For the sake of completeness, we recall that the solution proposed by Dorigo and colleagues in [41] differs for two inversions from the best Hamiltonian cycle proposed in [171], whereas genetic algorithms are able to obtain a minimal tour length equal to 424.635 non-dimensional units [82].

AddACO-V1. We start by analyzing how the performance of the V1 version of our AddACO is affected by variations in the values of α and β, which establish the trade-off between the two movement stimuli included in the ant decisional rule, see Eq. (5). In this setting, the size of the population of artificial insects is set equal to the number of cities of the TSP of interest (i.e.,

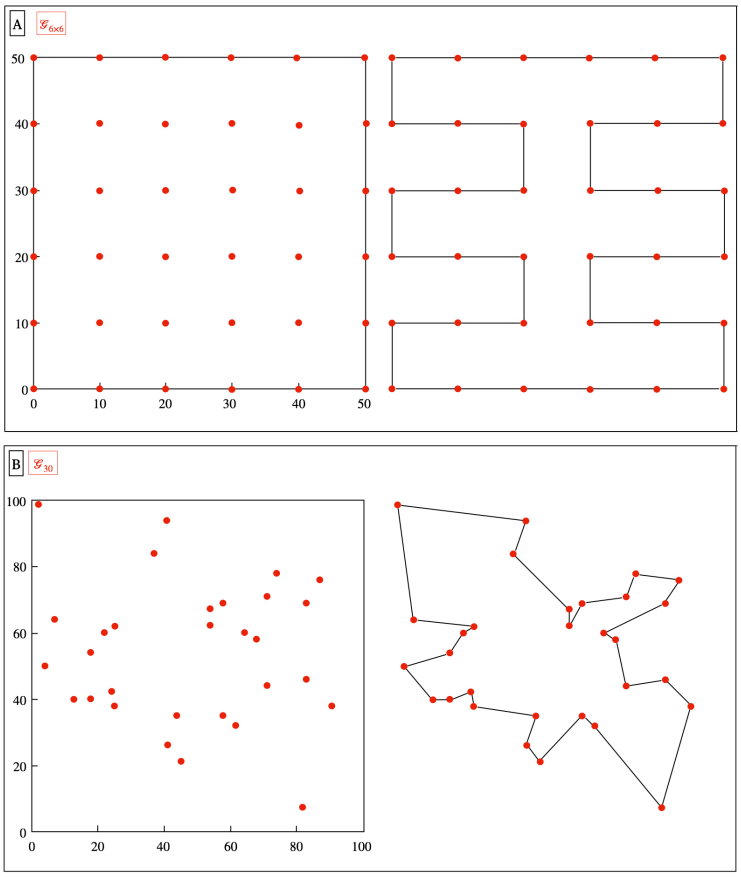

Figure 3: (A) The graph $\mathcal{G}_{6\times6}$ consists of a homogenous distribution of nodes, where the vertical and horizontal spacing is arbitrarily set equal to 10 non-dimensional units. In the right panel, we represent an optimal tour whose length is equal to 360 spatial units. Other optimal tours can be obtained by rotation of the reproduced one or by vertical/horizontal translation of one (or more) of its component edges. (B) Distribution of nodes in the case of graph $\mathcal{G}_{30}$, which is at the basis of the so-called *Oliver30* problem. In the right panel, we reproduce the optimal tour found by the Ant-Cycle version of the algorithm proposed by Dorigo and coworkers in [41], whose length is equal 423.74 non-dimensional units.

$M = N \in \{30, 36\}$), with an initially uniform distribution of agents (i.e., one for each node of the graph). Initially, there is no pheromone in the system: in this respect, when α is set equal to 1 the first tour of each ant simply consists of a random sequence of edges.

As shown in Table 1, in the case of $\mathcal{G}_{6\times6}$, the entire population of artificial insects constantly converges to minimal extension tours (scenario S_1) for intermediate values of α (i.e., $\in [0.4, 0.7]$). For small enough values of α (i.e., < 0.4), an optimal solution of the TSP is instead found in all runs by only a subgroup of agents, see the parameter region corresponding to scenario S_2 in the same Table 1. On the opposite, substantially large values of α (i.e., > 0.7) lead to stagnation (scenario S_3): in all experiments, the colony of ants in fact converges to equivalent tours that are not characterized by the shortest length. In particular, in such a parameter range, both the quality of the solutions and the robustness of the method (evaluated by V) decrease upon increments in α (rsp., decrements in β).

		$\mathcal{G}_{6\times6}$				$\mathcal{G}_{30}$			
α	β	BFT	$V(\%)$	UPB	S	BFT	$V(\%)$	UPB	S
0	1	**360**	0	no	S_2	578.96	7.23	no	S_4
0.1	0.9	**360**	0	no	S_2	538.96	6.32	no	S_4
0.2	0.8	**360**	0	no	S_2	499.87	5.35	no	S_4
0.3	0.7	**360**	0	no	S_2	436.75	3.81	no	S_4
0.4	0.6	**360**	0	yes	S_1	**423.74**	0	no	S_2
0.5	0.5	**360**	0	yes	S_1	**423.74**	0	no	S_2
0.6	0.4	**360**	0	yes	S_1	**423.74**	0	no	S_2
0.7	0.3	**360**	0	yes	S_1	430.68	8.76	yes	S_3
0.8	0.2	388.28	5.23	yes	S_3	445.56	14.23	yes	S_3
0.9	0.1	416.56	9.31	yes	S_3	524.56	18.13	yes	S_3
1	0	430.71	15.45	yes	S_3	578.89	23.09	yes	S_3

Table 1: Effect on the behavior of the AddACO-V1 of variations in α and β, which establish the trade-off between the two movement stimuli that affect the corresponding ant decisional rule, see Eq. (5).

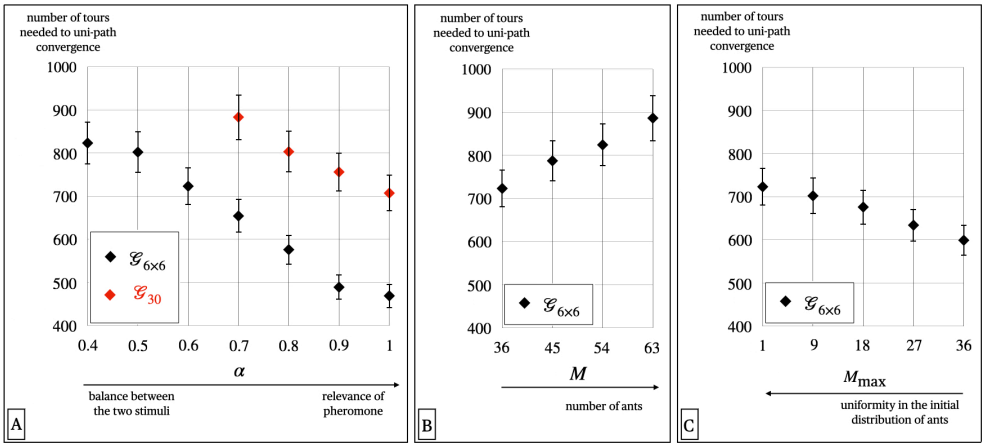

Figure 4: Number of tours needed to have a uni-path behavior in the case of the AddACO-V1 upon variations in selected parameters. In the plots, values are given as mean $\pm$ s.d. measured over the 20 numerical realizations run to derive the results shown in Tables 1, 2, and 3.

In the case of graph $\mathcal{G}_{30}$, the algorithm displays a significantly different parameter sensitivity, see again Table 1. In particular, minimal length tours are found in all runs for intermediate values of α (i.e., $\in [0.4, 0.6]$) but only by a limited number of agents (it is the algorithm behavior identified by S_2). A convergence to suboptimal solutions of the TSP (scenario S_3) is instead observed for $\alpha > 0.6$. Low values of α (i.e., < 0.4) finally result in the worst scenario S_4, as the artificial ants undergo uncorrelated movement and none of them is able to minimize its tour. In more details, in the case of $\mathcal{G}_{30}$, the quality of the solutions and the stability of the method drop for both high and low values of α (and β).

A further analysis of the results displayed in Table 1 allows to observe that the AddACO-

V1 converges to minimal length tours only in the case of the regular grid $\mathcal{G}_{6\times6}$. It is instead not able to couple uni-path behavior and optimal solution for the *Oliver30* distribution of cities. Somewhat consistently, the worst situation S_4 (i.e., no single ant is able to find a minimal tour) occurs only in the case of the graph $\mathcal{G}_{30}$. Furthermore, an optimal solution of the TSP is obtained for a significantly larger range of α-values in the case of $\mathcal{G}_{6\times6}$. In fact, for such a regular grid, even a substantially high relevance of the visibility guidance cue (established by low values of α) helps the probing activity of at least a subgroup of artificial agents, that are able to find a minimal length tour of the cities. This is due to the fact that the shortest tours of $\mathcal{G}_{6\times6}$ are formed by edges that connect each node with one of its closest ones (see Fig. 3 (A)). On the opposite, the (so-far found) optimal tour of $\mathcal{G}_{30}$ also includes edges that connect a city with another one that does not fall within its first-order neighborhood (cf. panel (B) in Fig. 3).

From Table 1, we can also notice that the uni-path behavior emerges in the case of both graphs for a high enough importance of the chemical signal. Large values of α (i.e., > 0.6) in fact imply that the artificial ants prefer to move along already explored edges and therefore the possibility to have a wide spectrum of different tours is reduced. In particular, the grid $\mathcal{G}_{6\times6}$ is more prone to the uni-path behavior, as convergence is observed also for $\alpha \in [0.4, 0.6]$, i.e., when the two migratory cues are almost balanced. It is however useful to remark that a too large relevance of the pheromone guidance stimulus prevents the algorithm to find optimal solutions: the agents need in fact to have the possibility to get rid of the past information gained by themselves of by their groupmates in order to achieve a more efficient scanning of movement options. Interestingly, for some of the tested parameter settings, the population of ants converges to different solutions in the different experiments, as it is captured by the couple of observed variables $V > 0$ and $UPB =$"yes".

The results relative to the graph $\mathcal{G}_{30}$ are consistent with those obtained in [41] by the Ant-Cycle version of the AS. In fact, Dorigo and colleagues show that their algorithm is not able to find optimal solutions to the TSP in the case of high relevance of trail, i.e., if it is coupled by a low relevance of the visibility stimulus. The worst scenario instead emerges in the opposite setting (i.e., poor importance of the pheromone signal and high relevance of visibility aspects). With the Ant-Cycle AS, minimal tours are finally found by small groups of agents for sufficiently balanced migratory cues.

The plot in Fig. 4 (A) shows that the number of tours needed by the colony of insects to reach a uni-path behavior in the case of our AddACO-V1 decreases upon increments in α. In particular, for any given value of α, the algorithm takes in average more iterations to converge in the case of $\mathcal{G}_{30}$ than in the case of $\mathcal{G}_{6\times6}$ (this is probably a consequence of the irregular distribution of the cities of the *Oliver30* topology). From the same plot, we can also conclude that:

- if it is the case, the algorithm converges within 1000 ant tours, independently from the graph of interest. This supports the choice of setting K_{MAX} equal to 5000 to infer a reasonable behavior of the proposed method;
- saturation effects characterize the uni-path behavior of this AddACO variant in the case of the graph $\mathcal{G}_{6\times6}$ for both $\alpha < 0.6$ and $\alpha > 0.8$.

Simulations are then run in order to assess the impact on the AddACO-V1 of variations in the

$\mathcal{G}_{6\times6}$					$\mathcal{G}_{30}$				
M	BFT	$V(\%)$	UPB	S	M	BFT	$V(\%)$	UPB	S
9	481.34	18.42	no	S_4	5	513.32	38.71	no	S_4
18	**360**	0	no	S_2	10	465.38	21.85	no	S_4
27	**360**	0	no	S_2	20	**423.74**	0	no	S_2
36	**360**	0	yes	S_1	30	**423.74**	0	no	S_2
45	**360**	0	yes	S_1	40	**423.74**	0	no	S_2
54	**360**	0	yes	S_1	50	**423.74**	0	no	S_2
63	**360**	0	yes	S_1	60	**423.74**	0	no	S_2

Table 2: Effect on the behavior of the AddACO-V1 of variations in the number of ants (M).

number of ants (M). In this respect, to have quasi-homogeneous starting configurations, we impose that:

- if $M \leq N$, there can not be more than one ant per city at $t = 0$;
- otherwise, i.e., if $M > N$, there can not be more than two ants per city at $t = 0$, considering that M is set to fall in the range $[1; 2N]$.

We also fix $\alpha = 0.6$ and $\beta = 0.4$, i.e., the parameter values giving rise to the best possible solutions in the case of both graphs (cf. Table 1).

As shown in Table 2, minimal length tours of the regular grid $\mathcal{G}_{6\times6}$ can be found by systems composed of at least 18 agents: however, convergence is restricted to the cases of sufficiently high population sizes (i.e., for $M \geq N = 36$). In particular, the number of iterations needed for the emergence of the uni-path behavior almost linearly increases (from ≈ 700 to ≈ 900) upon increments in the amount of artificial ants (cf. Fig. 4 (B)). In the case of $\mathcal{G}_{30}$, optimal solutions (without convergence) are instead obtained only by large enough systems of particles, i.e., for $M \geq 20$.

Interestingly, for both TSP instances, too low numbers of ants not only reduce the quality of the solutions but also significantly increase the variability of the numerical outcomes, as quantified by the values of V given in Table 2.

From these results, we can state that the ability of our algorithm to solve a TSP strongly relies on the number of exploratory agents, as a minimal population size is needed to find optimal tours. However, it is interesting to notice that in the case of $\mathcal{G}_{6\times6}$ such a value is substantially lower than the number of cities, whereas in the case of $\mathcal{G}_{30}$ the two quantities are comparable. The necessary amount of probing particles is therefore not only dictated by the number of cities to be visited but also by the regularity of their distribution.

Of course, the higher is the amount of insects the higher is the computational cost of the numerical approach, as demonstrated by Dorigo and colleagues by evaluating the so-called *one-ant cycles*, that are given as the average number of tours required to reach the optimum multiplied by the number of ants used in the different settings, see [41].

$\mathcal{G}_{6\times6}$					$\mathcal{G}_{30}$				
M_{max}	BFT	$V(\%)$	UPB	S	M_{max}	BFT	$V(\%)$	UPB	S
1	**360**	0	yes	S_1	1	**423.74**	0	no	S_2
9	**360**	0	yes	S_1	5	**423.74**	0	no	S_2
18	389.23	7.32	yes	S_3	20	476.45	11.37	no	S_4
27	425.46	15.23	yes	S_3	30	495.65	19.32	no	S_4
36	476.43	19.81	yes	S_3	40	524.52	24.32	yes	S_3

Table 3: Effect on the behavior of the AddACO-V1 of variations in the initial distribution of ants. In this respect, M_{max} quantifies the maximal number of agents that, at time $t = 0$, are allowed to be located on the same city, i.e., lower values of M_{max} indicated a more scattered initial distribution of artificial insects.

We now fix the number of ants equal to the number of cities of the graph of interest and vary their initial spatial pattern. In particular, we increase the maximal number of agents that, at time $t = 0$, are allowed to be located on the same (randomly chosen) city. If this value, hereafter labeled with M_{max}, is equal to 1, we have a completely homogenous initial distribution of ants (i.e., one per each node); on the opposite, if this value is equal to M, all artificial agents share the same starting city. Again, α is taken equal to 0.6 and β to 0.4.

Table 3 clearly shows that the population of ants is able to find minimal length tours only if it is initially scattered enough across the graph of interest. This algorithmic outcome occurs for both topologies of cities. In the case of the regular grid $\mathcal{G}_{6\times6}$, the uni-path behavior emerges independently from the ant distribution at $t = 0$: however, the number of iterations needed to have convergence decreases (to ≈ 600) upon increments in the initial ant clusterization, see panel (C) in Fig. 4. Similarly, in the case of the graph $\mathcal{G}_{30}$, convergence (to sub-optimal solutions) emerges after ≈ 850 tours only when all agents are initialized on the same node. It is in fact probable that ants starting from the same location, and subjected to the same stimuli, at the end perform the same set of equivalent tours. A substantial inhomogeneity in the initial insect distribution also affects the stability of the algorithm, as V is shown to increase with M_{max}, see again Table 3.

It is finally useful to underline that the above results are independent from the specific city/cities where the artificial ants are initially located, as we observed by means of supplementary simulations: an analogous behavior is shown to characterize the Dorigo's Ant-Cycle AS, at least in the cases of the problem *Oliver30* and of a 4×4 regular grid [41].

We now vary the number of ants that are allowed to release pheromone during movement. This model modification relies on the empirical observation that, in a wide spectrum of animal populations, only a subset of individuals has the task (and the ability) to guide the movement of the other groupmates. For instance, in bee colonies, a small fraction of insects (called *scouts*) inspects possible locations for the new nest to assess the quality, in terms of volume, aspect, size, and height of entrance. Then, they return to the rest of the swarm, which eventually takes off and compactly flies towards the chosen destination following their guidance. In particular, it has been hypothesized that the scout bees pilot the entire cloud of insects by producing the Nasonov pheromone, that indeed acts as a chemical trail [10]. In this respect, let us first identify with M_e

$\mathcal{G}_{6\times6}$					$\mathcal{G}_{30}$				
M_e	BFT	$V(\%)$	UPB	S	M_e	BFT	$V(\%)$	UPB	S
0	**360**	0	no	S_2	0	512.45	19.73	no	S_4
9	**360**	0	no	S_2	5	476.98	14.54	no	S_4
18	**360**	0	no	S_2	20	456.27	8.23	no	S_4
27	**360**	0	yes	S_1	30	**423.74**	0	no	S_2
36	**360**	0	yes	S_1	30	**423.74**	0	no	S_2

Table 4: Effect on the behavior of the AddACO-V1 of variations in the number of ants able to release the chemical trail (M_e).

($\leq M$) the number of such a sort of *elite* ants and modify Eq. (8) as follows:

$$q_e(t = zN) = q_e(t = (z-1)N) + \sum_{m=1}^{M_e} r(l_{c_z^m}) \mathbf{1}_{m \in S_e^z}, \tag{10}$$

where S_e^z defines the set of elite agents that have included the generic edge e in their last tour. For the sake of completeness, we recall that c_z^m, $l_{c_z^m}$, and $r(l_{c_z^m})$ are defined exactly as in Eqs. (8)-(9). All ants, regardless of their ability/inability to release the pheromone, are finally set to follow the movement probability law (5) characteristic of the variant V1 of our method.

Numerical tests are carried out by fixing $\alpha = 0.6$ (i.e., $\beta = 0.4$), setting the number of ants equal to the number of cities, with a homogenous initial distribution, and varying the amount of elite agents. In this respect, it is useful to underline that $M_e = 0$ implies that there is no ant able to release the chemical factor during its movement and therefore there is not a pheromone trail during the entire dynamics. All agents are indeed subjected only to the visibility stimulus, and therefore they typically opt for the closest (permitted) city, as in the case of $\alpha = 0$.

As shown in Table 4, in the case of the regular grid $\mathcal{G}_{6\times6}$, the number of elite ants does not affect the capacity of the algorithm to find an optimal solution, nor its robustness (V is always equal to 0): as already commented, minimal tours of this topology of cities are in fact formed by edges connecting first-order neighboring nodes and therefore they can be consistently discovered by at least few agents even in the absence of chemical trail. However, a high enough amount of elite individuals is necessary to have convergence of the entire population to the optimum: in particular, the number of tours needed to have the uni-path behavior is ≈ 850 for $M_e = 27$ and ≈ 700 for $M_e = M = 36$ (which corresponds to the case where all ants are allowed to release pheromone).

Minimal tours of $\mathcal{G}_{30}$ can be instead found only in the presence of sufficiently large numbers of elitists (i.e., for $M_e \geq 20$), that allow to have a relevant chemical pattern, see again Table 4. In particular, in the case of this graph, (i) the solution of the TSP is not coupled by a uni-path behavior of the algorithm, as none of the tested values of M_e results in its convergence, and (ii) the stability of the solutions increases (i.e., V decreases) with the number of elite ants.

In summary, the AddACO-V1 obtains the best possible results (that are strictly related to the graph of interest) only when the percentage of ants able to chemical release nearly exceeds the

		$\mathcal{G}_{6\times6}$				$\mathcal{G}_{30}$			
γ_{rand}	α_β	BFT	$V(\%)$	UPB	S	BFT	$V(\%)$	UPB	S
0	1	**360**	0	yes	S_1	**423.74**	0	no	S_2
0.25	0.75	**360**	0	no	S_2	498.34	15.34	no	S_4
0.5	0.5	438.78	12.45	no	S_4	513.56	34.45	no	S_4
0.75	0.25	556.87	45.45	no	S_4	613.45	49.34	no	S_4
1	0	674.43	67.54	no	S_4	778.45	68.54	no	S_4

Table 5: Effect on the behavior of the AddACO-V2 of variations in γ_{rand}, which establishes the relative importance of noise effects w.r.t. the other movement stimuli that affect ant decisional rule, see Eq. (6). In this respect, $\gamma_{rand} = 0$ results in the transition probability at the basis of the AddACO-V1. We recall that α_β is introduced in (11).

70% of the overall population. From experimental observations, it is however known that in most animal groups the fraction of elite individuals is substantially lower. The underlying reason is that, in ecological systems, leader agents transmit information to the rest of the group in several synergic ways. For instance, the scout bees, whose percentage in a swarm falls in the range $[3, 5]$, as evaluated in [23, 55], guide the other individuals also thanks to a particular flight strategy. As described in [106, 145, 146, 147], they are in fact observed to streak at high speed from the back of the swarm to its front: then, they either slowly fly back towards the rear edge of the cloud or stop and wait to be passed by the rest of the groupmates. In both cases, the dynamics are iterated until the population reaches the target destination (e.g., the new nest). Other hypotheses underlying movement synchronization mechanisms between leader and follower individuals in animal population involve directional alignment, long-range visual attraction and auditory signals.

We finally underline that the introduction of a subset of ants able to release pheromone implicates the inclusion of a sort of *elitist* strategy in the algorithm, which however differs from the one proposed in the literature [41, 42, 44].

AddACO-V2. We now test the exploratory capacity of the variant V2 of our ACO method on the two representative TSP instances $\mathcal{G}_{6\times6}$ and $\mathcal{G}_{30}$. In particular, we fix the size of the insect population equal to number of cities to be visited, with an initial homogenous distribution, and assign to all ants the ability to release the chemical trail. Furthermore, taking advantage of the above parametric analysis, we set the ratio β/α equal to 2/3. In this respect, recalling Eqs. (4) and (6), we have the following constraint:

$$\alpha+\beta+\gamma_{rand}=1 \implies \alpha+\frac{2}{3}\alpha+\gamma_{rand}=1 \implies \frac{5}{3}\alpha+\gamma_{rand}=1 \implies \alpha_\beta+\gamma_{rand}=1, \quad (11)$$

where the coefficient α_β gives a measure of the relative importance of the movement inputs at the basis of AddACO-V1 w.r.t. to the random contribution added in the variant V2 of our approach. For instances, low values of α_β imply high values of γ_{rand} and therefore high relevance of stochastic aspects in the ant exploratory behavior.

As shown in Table 5, the AddACO-V2 is substantially inferior to the V1 variant, in terms

		$\mathcal{G}_{6\times6}$				$\mathcal{G}_{30}$			
γ_{inert}	α_β	BFT	$V(\%)$	UPB	S	BFT	$V(\%)$	UPB	S
0	1	**360**	0	yes	S_1	**423.74**	0	no	S_2
0.25	0.75	389.32	12.31	no	S_4	476.51	15.74	no	S_4
0.5	0.5	434.98	12.98	no	S_4	512.13	15.98	no	S_4
0.75	0.25	513.56	13.12	no	S_4	561.45	16.31	no	S_4
1	0	592.41	13. 23	no	S_4	613.12	16.43	no	S_4

Table 6: Effect on the behavior of the AddACO-V3 of variations in γ_{inert}, which establishes the relative importance of inertia effects w.r.t. the other movement stimuli that affect ant decisional rule, see Eq. (7). In this respect, $\gamma_{inert} = 0$ results in the transition probability at the basis of the AddACO-V1. We recall that α_β is introduced in (11).

of both search capacity and robustness. In fact, even for sufficiently low values of γ_{rand}, the population of ants is constantly unable to find a minimal tour: in particular, the quality of the solutions decreases upon increments in the relevance of noise effects. The unique difference between the two topologies of cities is that, in the case of graph $\mathcal{G}_{6\times6}$, for $\gamma_{rand} = 0.25$ a subgroup of artificial agents is able to obtain an optimal solution of the problem.

A negative implication of a random term included in the ant decisional rule is also captured by Dorigo and colleagues with their Ant-Cycle AS, i.e., in the case of a 4×4 regular grid problem, see [41].

AddACO-V3. We finally analyze the performance of the third variant of our method, i.e., of the AddACO-V3. In particular, we study the effect of the inertial aspect, quantified by the coefficient γ_{inert}, cf. Eq. (7). In this respect, by recalling the considerations made in the above paragraph, we set $\alpha_\beta + \gamma_{inert} = 1$, with α_β defined as in Eq. (11). Again, the number of artificial insects is fixed equal to the size of the graph of interest and the ant population is initialized with a homogeneous pattern and entirely composed by pheromone-releasing individuals.

From Table 6, we can observe that, in the case of both representative graphs, inertia disrupts the possibility of the ant system to solve the TSP (and to converge to a uni-path behavior). For any $\gamma_{inert} \neq 0$, the numerical results obtained by the variant V3 of our method in fact fall in the worst scenario S_4, i.e., no artificial agent is able to perform a minimal length tour. In particular, increments in the relevance of the inertial contribution result in further decrements in the quality of the solutions. The underlying reason is that the optimal tours of the two topologies of cities include paths formed by completely misaligned edges and therefore the tendency to maintain a given direction of movement decreases the efficacy of the ant exploratory capacity. Interestingly, the persistence contribution in the ant decisional behavior does not have a significant impact on the stability of the algorithm, as the value of V is somehow constant despite variations in γ_{inert}.

Comments. From the above numerical results, it is clear that the variant V1 of our AddACO is significantly superior than the other two versions. It is in fact the solely able to find an optimal tour of the two representative distributions of cities, i.e., if a proper set of parameters is employed. In this respect, we can state that the addition to the canonical pair of ant decision variables (i.e.,

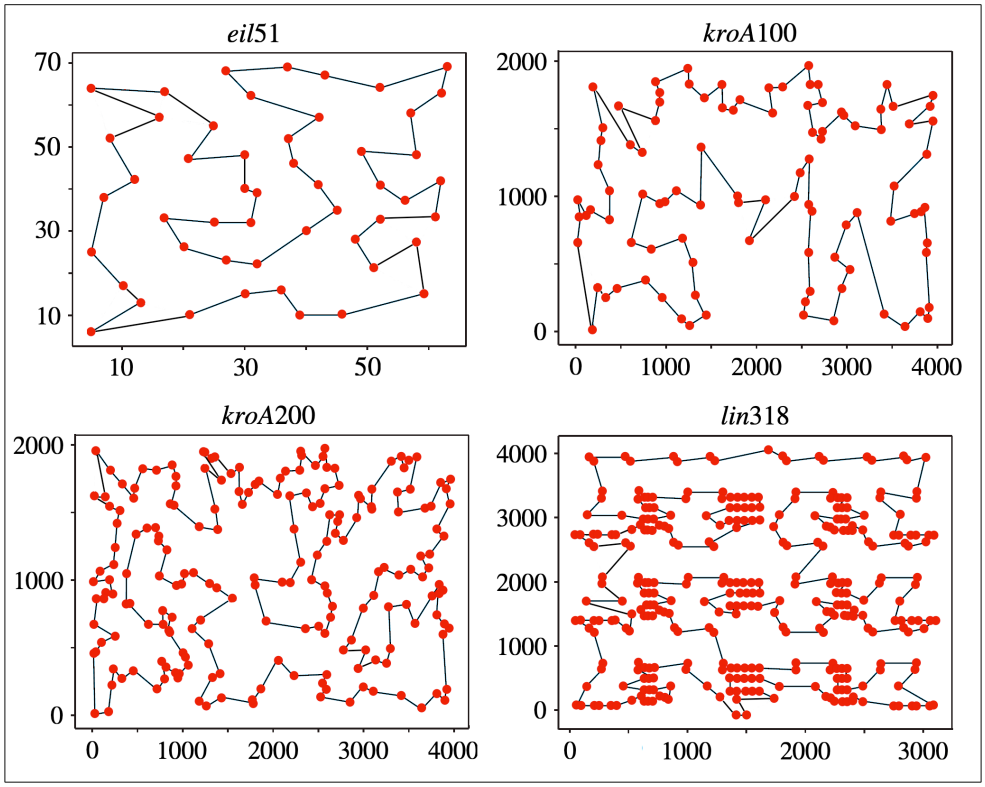

Figure 5: Distribution of cities in the case of the four TSP instances selected to further evaluate the performance of our method, also in comparison with the canonical Ant-Cycle AS. In each case, we depict the minimal length path obtained with the AddACO-V1.

pheromone trail and inter-nodal proximity) of the proposed movement stimuli does not increase the search behavior of the insect population. It is also interesting to notice that the ability of all AddACO variants to find an optimal tour of a given graph is constantly coupled by a null variability of the solutions.

4.2. Application of the AddACO-V1 to large-scale graphs

The AddACO-V1 is then applied to large-scale benchmark TSP instances and compared, in terms of performance, to the Ant-Cycle version of the AS proposed by Dorigo and colleagues.

In particular, in the forthcoming tests, the variant V1 of our method is run with the best-performing coefficients found in the previous Subsection 4.1, i.e., $\alpha = 0.4$ and $\beta = 0.6$ (being the pheromone release rate equal to 1). For the Ant-Cycle AS, we instead use the parameter setting recommended in [41], i.e., $\alpha = 1$ and $\beta = 5$ with the chemical release rate and evaporation constant equal to 1 and 0.5, respectively. In all cases, the size of the ant population matches the number of nodes of the graph of interest, with an initial homogenous distribution of artificial insects (i.e., one per each city).

For a fair comparison, the two algorithms are both run for $K_{\text{MAX}} = 5000$ blocks of iterations (i.e., the ants are allowed to perform 5000 tours to find the best-possible solution of the problem), while the results are averaged over 20 independent trials. In this respect, the performance of the

		AddACO-V1			AntCycle AS		
TSP	BKT	BFT	$E(\%)$	$V(\%)$	BFT	$E(\%)$	$V(\%)$
eil51	426	462	8.45	9.83	449	5.39	12.53
kroA100	21 282	23 125	8.66	10.32	23 308	9.52	12.32
kroA200	29 368	32 832	11.80	10.63	31 846	8.44	15.65
lin318	42 029	48 742	15.97	11.76	47 659	13.40	14.33

Table 7: Performance comparison between the AddACO-V1 and the Ant-Cycle AS both applied on the same set of representative TSP instances.

tested methods is quantified by the observable variables BFT and V, defined at the beginning of Section 4, and by the relative error

$$E = \frac{BFT - BKT}{BKT} * 100, \tag{12}$$

where BKT is the length of best-known tour of the graph of interest taken from the literature.

The proposed experiments are conducted on symmetric TSP instances selected within the TSPLIB Benchmark library[2], which contains a large collection of graphs derived from theoretical studies and practical applications. We recall that the TSPLIB repository labels a graph by a unique code composed of letters and digits, the latter indicating the number of nodes. In particular, the following distributions of cities are hereafter taken into account: *eil51, kroA100, kroA200,* and *lin318*, see Fig. 5. Their sizes fall in a sufficiently large range to allow a significant analysis of the exploratory ability of our method (even in comparison with the canonical Dorigo's version).

From Table 7, we can first observe that both algorithms constantly fail to find the best tour of the proposed TSPs. However, in all cases, the AddACO-V1 shows a competitive performance, in terms of exploration capacity, compared to the Ant-Cycle AS, as the two methods obtain good enough solutions. In particular, the discrepancy between the minimal tour obtained by the AddACO-V1, shown in Fig. 5, and the best-known one moderately increases with the size of the graph but it does not exceed the 16%, as evaluated by the quantity E. In this respect, it is interesting to notice that, in the case of the *kroA100* instance, our algorithm outperforms the Ant-Cycle AS.

For all considered TSPs, the stability of the solutions found by the AddACO-V1 is slightly higher than the corresponding quantity evaluated for the Ant-Cycle AS. In particular, our method is characterized by a variability V that constantly falls in the (approximated) range $[9; 12]$, being also independent from the scale of the graph of interest: this is indicative of a substantial robustness.

We finally remark that neither the AddACO-V1 nor the Ant-Cycle AS are characterized by a uni-path behavior in the experiments carried out in this Subsection.

[2]http://www.iwr.uni-heidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB.html

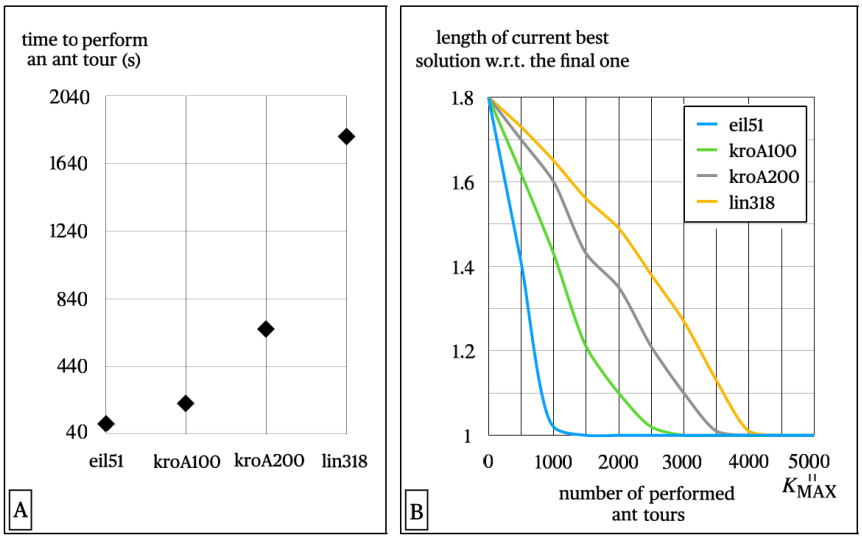

Figure 6: (A) Time needed by our AddACO-V1 to process a single ant tour in the case of the four TSP instances taken as benchmark tests in Subsection 4.2. (B) Evolutionary process of the current best solution obtained by our method in the same set of experiments. In both graphs, the values are taken from a single representative numerical run. Little variations are in fact observed by taking the average values over a higher amount of realizations.

We now analyze the training time of our method, evaluated as the actual number of seconds needed by the algorithm to perform a single ant tour in the case of the four above-introduced graphs. We recall that a tour is composed of N basic step, being N the number of cities to be visited. As it can be seen from Fig. 6 (A), the time consumption of the AddACO-V1 has a square relationship (i.e., a parabolic trend) with the scale of the TSP to which it is applied. In particular, to complete an ant cycle, it takes 49 s for the *eil51*, 201 s for the *kroA100*, 679 s for the *kroA200*, and 1791 s for the *lin318*. In this respect, to perform the entire amount of allowed tours (i.e., equal to $K_{\text{MAX}} = 5000$), our method approximatively takes $2.45{\cdot}10^5$ s for the *eil51*, $1{\cdot}10^6$ s for the *kroA100*, $3.3{\cdot}10^6$ s for the *kroA200*, and $8.9{\cdot}10^6$ s for the *lin318*, which is indicative of a linear relationship between the computational time and the amount of performed cycles.

From Table I in [41], we can observe that an ant tour of a 49-city graph requires nearly 42 s to the Ant-Cycle AS, which is close to the corresponding value recorded for our method in the case of the *eil51* problem, i.e., in the case of a network with a comparable size. This is not surprising, since the main numerical cost of both algorithms comes from the computation of the probabilities at the basis of the corresponding ant decisional rules, which differ only for their characteristic form (i.e., additive vs multiplicative) and not for the number of required operations.

To have a further confirmation of this aspect, we run the other two variants of our method on the four benchmark TSP instances introduced in this Subsection. We find that the time needed to perform a single ant tour increases (in average) of the 23% in the case of the AddACO-V2 and of the 28% in the case of the AddACO-V3. The underlying reason is that these two versions of our algorithm include three behavioral stimuli in the transition probability, see Eqs. (6) and (7), that increase the amount of operations to be performed at any given construction step. In particular, the variant V3 is computationally more expensive than the variant V2, since the

evaluation of angles between arcs represents a numerical overhead of the evaluation of random functions uniformly distributed over a given interval.

We finally study the convergence speed of our AddACO-V1, that is quantified by the evolution of the ratio (in terms of length) between the best solution obtained by the algorithm at a given iteration and the best one found at the end of its run (i.e., after the permitted $K_{\mathrm{MAX}} = 5000$ tours). This quantity is a measure of the improvement of the solution quality through iterations. From the plot in Fig. 6 (B), we can observe that, in the case of all four TSP instances taken into account, the length of the current best solution linearly decreases over time, with a rate of decrement that depends on the scale of the graph of interest. In particular, the larger is the number of cities the lower is the convergence speed of our method. In more details, to obtain the best-possible solution, the population of ants needs nearly 1500 tours in the case of the *eil51*, 3000 in the case of the *kroA100*, 3600 in the case of the *kroA200*, and almost 4200 in the case of the *lin318*. From these data, we can conclude that increments in the quality of the solutions obtained by our method can not be obtained even by setting K_{MAX} larger than 5000, at least in the considered cases.

Comments. The above numerical results allow us to state that the additive form here proposed for the ant decisional rule does not have a negative impact on the exploratory ability of the insect population, in terms both of solution quality and of computational time and cost. In all cases taken here into account, the performances of the AddACO-V1 are in fact competitive with those of the Ant-Cycle AS, i.e., with those of a similar method that employs the canonical multiplicative form of the transition rule and that is not enriched by any further optimization strategy.

5. Conclusions

A wide spectrum of numerical algorithms has been developed over years to tackle TS and more in general CO problems: they can be classified as either *complete/exact* or *approximate* methods. The former group includes algorithms that are in principle able to find an optimal solution for any finite size instance of a CO problem [124, 130]. However, these approaches may require computation costs and times that are too high for practical purposes (as in the case of NP-hard problems [62]). Approximate algorithms represent a reasonable compromise in this respect, as they allow to obtain suboptimal solutions of a given CO problem but with a significantly reduced amount of computation.

One of the most famous family of approximate methods employed to solve TSPs is represented by the Ant Colony Optimization (ACO), which takes inspiration from the collective foraging behavior of these insects. It is in fact widely shown that a population of interacting artificial agents, by communication via pheromone trails, initiates an autocatalytic process that, in conjunction with a greedy force, results in the ability to find very good, often optimal, tours of a given distribution of points (cities). In particular, as commented in [14], even if no tour becomes unfeasible, bad tours become highly improbable, as ACO algorithms search only in the neighborhood of good solutions.

ACO methods can be considered relevant examples in the field of the swarm intelligence (SI), which consists in the design of multi-agent systems whose collective behavior is shaped on social rules and interactions characterizing selected animal groups, such as flocks of birds, schools of

fishes, bee clouds, and so on [17, 18, 22, 90, 112]. From the perspective of the operation research (OR), ACO algorithms can be instead classified in the class of meta-heuristics [15, 65, 66], as formalized by Dorigo and colleagues firstly in [43] and then in [45].

In this paper, we propose the AddACO, a modified version of the algorithm presented by Dorigo and colleagues in [41]. In particular, in our approach, the probabilistic law that, at any solution construction step, underlies ant decision on the next destination has an *additive* form, i.e., it is given by a convex linear combination of selected migratory stimuli, each of them weighted by a coefficient that defines its relative (and independent) effect. The unitary sum given to these weights also allows to avoid an implausible simultaneous minimization/maximization of the entire set of behavioral traits taken into account.

As already claimed in the Introduction, the proposed method is not developed with the aim to find a superior approach w.r.t. those already present in the literature. This aspect is confirmed by the fact that we do not include in our numerical framework strategies to increase its exploratory ability and/or to optimize the computational cost (such as those to improve local search or parametrization). Rather, the AddACO is simply an *alternative* ACO method characterized by a *different* transition probability whose additive structure, according to us, in principle represents a conceptual improvement w.r.t. to the canonical multiplicative one.

Three different variants of the AddACO are then proposed, each characterized by a specific set of decisional variables assumed to affect ant movement. In this respect, by reviewing the numerical results shown in Section 4, we can conclude that:

- the AddACO-V1 outperforms the other two variants of our method;
- in more details, the best possible results obtained by the AddACO-V1 are found when (i) there is a substantial balance between the two migratory stimuli (i.e., the pheromone trail and the visibility), (ii) the number of ants is close to the number of cities to be visited, and (iii) the insects have an homogenous initial distribution, with a large majority of them able to chemical release during movement;
- for reasonably large TSP instances, the AddACO-V1 can consistently (i.e., with a low variability) find solutions of a sufficiently high quality. In particular, in all considered cases, its performances are close to those of the classical Ant-Cycle AS. The computational time is also similar between the two approaches.

The numerical outcomes shown throughout this paper allow us to claim that the additive form of the transition probability proposed here does not negatively impact on the ant exploratory behavior.

The AddACO-V1 can be indeed considered a promising method: however, its performance is still not at the level of most state-of-the-art methods employed to solve TSPs. In this respect, Tables 8 and 9 collect the results obtained in the solution of the four benchmark TSP instances introduced in the previous section by a wide range of prominent algorithms, that belong both to the ACO approach and to other families of heuristics and meta-heuristics (for the sake of reader's convenience, we remark that the methods included for comparison purposes in the two Tables have been commented and reviewed, at least in their main aspects, in Section 1). In all

TSP (BKT)	AddACO-V1	AHACO [48]	ACOAV [148]	PACO [70]	DEACO [49]	HAACO [162]	PSO-ACO [113]	SOS-ACO [169]	ACO-LO [169]	ACSFA [8]
eil51 (426)	462 (8.45)	428.87 (0.67)	**426** (0.00)	**426** (0.00)	**426** (0.00)	**426** (0.00)	**426** (0.00)	**426** (0.00)	430 (0.94)	428 (0.47)
kroA100 (21 282)	23 125 (8.66)	21 285.4 (0.02)	**21 282** (0.00)	**21 282** (0.00)	**21 282** (0.00)	**21 282** (0.00)	21 301 (0.09)	**21 282** (0.00)	21 892 (2.87)	21 396 (0.54)
kroA200 (29 368)	32 832 (11.80)	29 460 (0.31)	**29 368** (0.00)	29 533 (0.56)	**29 368** (0.00)	29 483 (0.39)	29 468 (0.34)	29 413 (0.15)	30 115 (2.54)	-
lin318 (42 029)	48 742 (15.97)	42 250.7 (0.53)	42 203 (0.41)	-	-	-	-	42 473 (1.06)	44 849 (6.71)	43 061 (2.46)

Table 8: Performance comparison between the AddACO-V1 and some recent ACO-based methods in the case of the four representative TSP instances taken as benchmark tests in Section 4.2. In all cases, we indicate the best-found tour (BFT) obtained by the corresponding method with the percentage error (E), defined in Eq. (12), between parentheses.

TSP (BKT)	AddACO-V1	DMSO [2]	DLSO [33]	DSOS [53]	VTPSO [3]	SSABC [91]	GA-MARL+NICH-LS [6]	HGA [165]
eil51 (426)	462 (8.45)	428.86 (0.67)	428.87 (0.67)	427 (0.23)	429.51 (0.82)	427 (0.23)	**426** (0.00)	428.9 (0.68)
kroA100 (21 282)	23 125 (8.66)	21 298.21 (0.08)	21 285.44 (0.02)	**21 282** (0.00)	**21 307.44** (0.12)	**21 282** (0.00)	**21 282** (0.00)	21 285.4 (0.16)
kroA200 (29 368)	32 832 (11.80)	30 481.35 (3.79)	29 519.83 (0.52)	29 477 (0.37)	30 602.81 (4.20)	29 450 (0.28)	29 435 (0.23)	29 369.4 (0.00)
lin318 (42 029)	48 742 (15.97)	44 118.66 (4.97)	-	42 201 (0.41)	44 724.38 (6.41)	-	42 255 (0.54)	42 624.3 (1.42)

Table 9: Performance comparison between the AddACO-V1 and some state-of-the-art heuristics and meta-heuristics in the case of the four representative TSP instances taken as benchmark tests in Section 4.2. In all cases, we indicate the best-found tour (BFT) obtained by the corresponding method with the percentage error (E), defined in Eq. (12), between parentheses.

cases, we indicate the best-found tour (BFT) obtained by the corresponding algorithm and the percentage error (E) evaluated w.r.t. the best-known tour (BKT), as defined in Eq. (12). The results are taken from published works, as indicated in the first row of the two Tables. The absence of an available outcome is marked with "-". The data in Tables 8 and 9 confirm that the solutions obtained by the AddACO have a substantially lower quality than those found both by more sophisticated ACO-based methods and by other recent heuristics and meta-heuristics successfully employ to solve TSP instances. This is not surprising as also the naive Ant-Cycle AS was consistently outperformed by subsequently proposed algorithms.

In this respect, it would be surely interesting to improve the AddACO by the inclusion of local search strategies (e.g., k-Opt techniques) and/or by the hybridization with other heuristics/meta-heuristics, i.e., in order to increase its exploratory capacity in a perspective of a fair comparison with selected state-of-the-art and high-performing methods.

A further possible future perspective may then amount in the application of an improved AddACO to other classes of optimization problems. For instance, as commented in [14], bioinformatic and biomedical researchers show an increasing interest in ACO-based methods, as provided by their application to problems in the fields of protein folding [150, 151], of DNA multiple sequence alignment [121], and of histocompatibility prediction, which plays a relevant role for instance in vaccine developments [89]. Our method, if properly developed, could in principle address also a wide spectrum of practical issues in industry, such as those already cited and relative to sequential ordering [58], project scheduling limited by constrained resources [119], and optimization of shop opening [16].

Appendix A. Asymptotic behavior of the AddACO-V1

A common theoretical problem relative to ACO algorithms concerns their large-time behavior. In this respect, an asymptotic analysis of these methods has to consider two aspects:

1. the ability of the algorithm to find, at least once, an optimal solution of the TSP of interest, sometimes called *convergence in value* [14];

2. the probability that the algorithm, starting from a given iteration, constantly generates the same optimal solution, a phenomenology referred to as *convergence in solution* [14].

In this respect, the above-defined scenarios S_1 and S_2 certainly imply a convergence in value of our method. The former scenario reasonably captures a convergence in solution as well.

The first asymptotic analysis concerning an ACO algorithm is proposed by Gutjahr in [73, 74]. Dorigo and colleagues then provide a complete study on the long-run behavior of a larger spectrum of this family of methods [45, 157]. We indeed build on their dissertation and prove that the variant V1 of our AddACO algorithm (i.e., the best performing one) is guaranteed to converge in value regardless the specific distribution of cities if the population of artificial ants is not only subjected to pheromone guidance cues. In this respect, we have the following theoretical result.

Theorem 1. *Let $P(t = zN)$ be the probability that the algorithm finds an optimal solution of the TSP of interest at least once within the first z blocks of iterations, being $z \in \mathbb{N}, z > 0$. In other words, $P(t = zN)$ is the probability that at least a single artificial ant performs a minimal length tour of the graph within its first z attempts. If the virtual agents move according to Eq. (5), with $\alpha \neq 1$, and the pheromone update law is given by Eqs. (8)-(9), then it asymptotically holds that:*

$$\lim_{t \to +\infty} P(t) = \lim_{k \to +\infty} P(zN) = 1. \tag{A.1}$$

Proof. At any given iteration t and ant m, each feasible movement option is characterized by a strictly positive probability $p_{\min} > 0$. For any pair of representative nodes n_i and $n_j \in V^m(t)$, from Eq. (5), we have in fact that:

$$p_{n_i \to n_j}(t) \geq \beta \frac{\left(\max\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} l_{e_{ik}} \right)^{-1}}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} (l_{e_{ik}})^{-1}} = p_{\min}. \tag{A.2}$$

To derive the above inequality, we have considered the "worst case" setting, i.e., given by $l_{e_{ij}} = \max\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} l_{e_{ik}}$ coupled by one (or more) of the following conditions: $q_{e_{ij}} = 0$, $\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} q_{e_{ik}(t)} = +\infty$ (i.e., $\max\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} q_{e_{ik}(t)} = +\infty$), or $\alpha = 0$. We also remark that the hypothesis of the Theorem implies $\beta \neq 0$ and that the lengths of all edges are finite and larger than 0 (i.e., there are not overlapped cities). All candidate solutions of the TPS of interest (i.e., all possible tours of the cities), including the optimal ones $c^* \in \mathcal{C}^*(\mathcal{G})$, can be then generated with a probability that is bounded below by the value $P_{\min} \geq p_{\min}^N > 0$, being N the number of nodes of $\mathcal{G}$ or, equivalently,

the number of edges to be traversed by the ants to complete a tour. Let us now define by $\overline{P}(t = zN) = 1 - P(t = zN)$ the probability that at the z-th block of iterations no artificial insect has found an optimal solution of the TSP. We indeed have that:

- for $z = 1$, $\overline{P}(t = N) = 1 - P_{\text{min}} \leq 1 - p_{\text{min}}^N$;
- for $z = 2$, $\overline{P}(t = 2N) = (1 - P_{\text{min}})^2 \leq (1 - p_{\text{min}}^N)^2$
- by iteration, for any $z > 0$, $\overline{P}(t = kN) = (1 - P_{\text{min}})^k \leq (1 - p_{\text{min}}^N)^k$.

The above inequalities imply that:

$$1 - (1 - p_{\text{min}}^N)^k \leq 1 - (1 - P_{\text{min}})^k = 1 - \overline{P}(t = zN) = P(t = zN) \leq 1 \tag{A.3}$$

By observing that $(1 - p_{\text{min}}^N) \leq 1 \implies \lim_{k \to +\infty} (1 - p_{\text{min}}^N)^k = 0$, we can conclude that:

$$\lim_{z \to +\infty} P(t = zN) = 1. \tag{A.4}$$

□

Observations and comments:

- Theorem 1 states that the AddACO-V1 is asymptotically able to find an optimal solution of a given TSP, i.e., if its assumptions hold. However, it does not say anything about the computational cost and time needed, which can be substantially large. As remarked in [157], a similar limitation applies to most studies relative to the long-run convergence of numerical methods, such as those for simulated annealing [75, 140];
- the proposed analytical results do not contradict the simulation outcomes shown in the previous Section 4. In several settings the population of artificial ants has been unable to perform minimal tours even in the case of $\alpha \neq 1$: however, this is due to the fact that we have opted to run the algorithm for $K_{\text{MAX}} = 5000$ blocks of iterations, i.e., a value sufficiently high to reasonably test the efficacy of our method (as also commented in [41]) but too small to infer its asymptotic behavior;
- as demonstrated in [157], Theorem 1 can be applied to any ACO algorithm for which the probability of constructing any candidate solution is strictly positive. This is the reason why we have to restrict our analytical results to the case $\alpha \in [0, 1)$. Otherwise, for a given iteration t, ant m, and representative pair of nodes n_i and $n_j \in V^m(t)$, we could have:

$$p_{\text{min}} = \frac{\min\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} q_{e_{ik}}(t)}{\sum\limits_{\substack{e_{ik} \in E \\ n_k \in V^m(t)}} q_{e_{ik}}(t)} = 0. \tag{A.5}$$

In fact, our version of the algorithm neglects pheromone evaporation (cf. Eq. 8): this implies that the quantity of chemical substance may be not asymptotically bounded in one or more edges and that therefore the denominator of the fraction in (A.5) may go to infinity. The possibility that the numerator is zero may be instead easily avoided by employing a different pheromone initialization w.r.t. the condition proposed in Section 4, i.e., it would be sufficient to assign to each edge of the graph of interest a positive quantity of chemical substance;

- the thesis of Theorem 1 holds also in the case of the variants V2 and V3 of our method, i.e., those based on the ant transition laws given in Eqs. (6) and (7), respectively, if we further assume that $\gamma \in \{\gamma_{rand}; \gamma_{inert}\} \neq 1$. We can not in fact guarantee that the third contribution included in those probability functions is strictly positive.

References

[1] Y. Ahmad, M. Ullah, R. Khan, B. Shafi, A. Khan, M. Zareei, A. Aldosary, E. M. Mohamed, SiFSO: Fish Swarm Optimization-based technique for efficient community detection in Complex networks, Complexity 2020 (2020) 6695032.

[2] M. Akhand, S. I. Ayon, S. Shahriyar, N. Siddique, H. Adeli, Discrete Spider Monkey Optimization for Travelling Salesman Problem, Appl. Soft Comput. 86 (2019) 105887.

[3] M. A. H. Akhand, S. Akter, M. A. Rashid, S. B. Yaakob, Velocity tentative PSO: An optimal velocity implementation based Particle Swarm Optimization to solve Traveling Salesman Problem, IAENG Int. J. Comput. Sci. 42 (2015) 221–232.

[4] M. A. H. Akhand, Z. Jannat, K. Murase, Capacitated vehicle routing problem solving using adaptive sweep and velocity tentative PSO, International Journal of Advanced Computer Science and Applications 8 (2017).

[5] I. M. Ali, D. Essam, K. Kasmarik, A novel design of differential evolution for solving discrete Traveling Salesman Problems, Swarm and Evolutionary Computation 52 (2020) 100607.

[6] M. M. Alipour, S. N. Razavi, M. R. F. Derakhshi, M. A. Balafar, A hybrid algorithm using a genetic algorithm and multiagent reinforcement learning heuristic to solve the Traveling Salesman Problem, Neural Comput. Appl. 30 (2017) 2935–2951.

[7] D. Anghinolfi, A. Boccalatte, M. Paolucci, C. Vecchiola, Performance evaluation of an adaptive Ant Colony Optimization applied to single machine scheduling, Proc. of Asia-Pacific Conference on Simulated Evolution and Learning (2008) 411–420.

[8] M.K.A. Ariyaratne, T.G.I. Fernando, S. Weerakoon, A self-tuning firefly algorithm to tune the parameters of Ant Colony System, Int. J. Swarm. Intell. 3 (2018) 309–331.

[9] P. Baniasadi, M. Foumani, K. Smith-Miles, V. Ejov, A transformation technique for the clustered generalized Traveling Salesman Problem with applications to logistics, European J. Oper. Res. 285 (2020) 444–457.

[10] M. Beekman, R. L. Fathke, T. D. Seeley, How does an informed minority of scouts guide a honeybee swarm as it flies to its new home?, Animal Behaviour 71 (2006) 161–171.

[11] J.L. Bentley, Fast algorithms for geometric Traveling Salesman Problems, ORSA Journal on Computing 4 (1992) 387–411.

[12] D. P. Bertsekas, J. N. Tsitsiklis, C. Wu, Rollout algorithms for combinatorial optimization, J. Heuristics 3 (1997) 245–262.

[13] B. Bilchev, I. C.Parmee, The ant colony metaphor for searching continuous design spaces, in: T. C. Fogarty (Ed.), Proceedings of the AISB Workshop on Evolutionary Computation, Lecture Notes in Comput. Sci., vol. 993, Springer Berlin, 1995, pp. 25–39.

[14] C. Blum, Ant Colony Optimization: Introduction and recent trends, Physics of Life Reviews 2 (2005) 353–373.

[15] C. Blum, A. Roli, Metaheuristics in combinatorial optimization: Overview and conceptual comparison, ACM Comput. Surveys 35 (2003) 268–308.

[16] C. Blum, Beam-ACO-hybridizing ant colony optimization with beam search: An application to open shop scheduling, Computers & Operations Res 32 (2005), 1565–1591.

[17] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm intelligence: From natural to artificial systems, Oxford University Press New York, 1999.

[18] E. Bonabeau, M. Dorigo, G. Theraulaz, Inspiration for optimization from social insect behavior, Nature 406 (2000), 39–42.

[19] H. M. Botee, E. Bonabeau, Evolving Ant Colony Optimization, Adv. Complex. Syst. 1 (1998) 149–159.

[20] A. Bouzidi, M. E. Riffi, Discrete Cat Swarm Optimization to resolve the Traveling Salesman Problem, Int. J. Adv. Res. Comput. Sci. Softw. Eng. 3 (2013) 13–18.

[21] B. Bullnheimer, R. Hartl, C. Strauss, A new rank-based version of the Ant System: A computational study, Central European J. Operations Res. Econom. 7 (1999) 25–38.

[22] M. Campos, E. Bonabeau, G. Theraulaz, J.-L. Deneubourg, Dynamic scheduling and division of labor in social insects, Adapt. Behavior 8 (2000) 83–96.

[23] I. D. Couzin, J. Krause, N. R. Franks, S. A. Levin, Effective leadership and decision-making in animal groups on the move, Nature 433 (2005) 513–516.

[24] G. Campuzano, C. Obreque, M. M. Aguayo, Accelerating the Miller–Tucker–Zemlin model for the asymmetric Traveling Salesman Problem, Expert Syst. Appl. 148 (2020) 113229.

[25] O. Cheikhrouhou, I. Khoufi, A comprehensive survey on the multiple Traveling Salesman Problem: Applications, approaches and taxonomy, Comput. Sci. Rev. 40 (2021) 100369.

[26] H. Chen, W. Li, X. Yang, A whale optimization algorithm with chaos mechanism based on quasi-opposition for global optimization problems, Expert Syst. Appl. 158 (2020) 113612.

[27] M.Y. Cheng, D. Prayogo, Symbiotic Organisms Search: A new meta-heuristic optimization algorithm, Comput. Struct. 139 (2014) 98–112.

[28] S. Climer, W.X. Zhang, Cut-and-solve: An iterative search strategy for combinatorial optimization problems, Artificial Intelligence 170 (2006) 714–738.

[29] A. Colorni, M. Dorigo, and V. Maniezzo. Distributed optimization by Ant Colonies, in: Proceedings of the First European Conference on Artificial Life (ECAL 91), pages 134–142. Elsevier, 1991.

[30] M. Cornu, T. Cazenave, D. Vanderpooten, Perturbed decomposition algorithm applied to the multi-objective Traveling Salesman Problem, Comput. Oper. Res. 79 (2017) 314–330.

[31] G. A. Croes, A method for solving Traveling Salesman Problems, Oper. Res. 6 (1958) 791–812.

[32] F. Dahan, K. El Hindi, H. Mathkour, H. AlSalman, Dynamic Flying Ant Colony Optimization (DFACO) for solving the Traveling Salesman Problem, Sensors 19 (2019) 1–28.

[33] Z. Daoqing, J. Mingyan, Parallel Discrete Lion Swarm Optimization algorithm for solving Traveling Salesman Problem. J. Syst. Eng. Electron. 31 (2020) 751–760.

[34] J. L. Deneubourg, S. Aron, S. Goss, J. M. Pasteels, The self-organizing exploratory pattern of the Argentine ant, J. Insect Behaviour 3 (1990) 159–68.

[35] J. L. Denebourg, J. M. Pasteels, J. C.Verhaeghe, Probabilistic behaviour in ants: A strategy of errors?, J. Theor. Biol. 105 (1983) 259–271.

[36] J. L. Denebourg, S. Goss, Collective patterns and decision-making, Ethology, Ecology & Evolution 1 (1989) 295–311.

[37] M. Deudon, P. Cournut, A. Lacoste, Y. Adulyasak, L. M. Rousseau, Learning heuristics for the TSP by policy gradient, in: Integration of Constraint Programming, Artificial Intelligence, and Operations Research, Springer, Cham, 10848 (2018) 170–181.

[38] A. Delevacq, P. Delisle, M. Gravel, M. Krajecki, Parallel Ant Colony Optimization on graphics processing units, Journal of Parallel and Distributed Computing 73 (2013) 52–61.

[39] I. M. Diaaeldin, S. H. A. Aleem, A. El-Rafei, A. Y. Abdelaziz, M. Calasan, Optimal network reconfiguration and distributed generation allocation using Harris Hawks Optimization, in: Proc. of 2020 24th International Conference on Information Technology (IT), Zabljak, Montenegro, pp. 1-6, 2020.

[40] X. Dong, H. Zhang, M. Xu, F. Shen, Hybrid genetic algorithm with variable neighborhood search for multi-scale multiple bottleneck Traveling Salesmen Problem, Future Generation Computer Systems 114 (2021) 229–242.

[41] M. Dorigo, V. Maniezzo, A. Colorni, Positive feedback as a search strategy, Technical Report 91-016, Dip. Elettronica, Politecnico di Milano, Italy 1991.

[42] M. Dorigo. Optimization, Learning, and Natural Algorithms (in Italian). PhD thesis, Dip. Elettronica, Politecnico di Milano, 1992.

[43] M. Dorigo, G. Di Caro, L. M. Gambardella, Ant algorithms for discrete optimization, Artificial Life 5 (1999) 137–172.

[44] M. Dorigo, V. Maniezzo, A. Colorni, Ant System: Optimization by a colony of cooperating agents, IEEE Trans. Syst. Man. Cybernet Part B 26 (1996) 29–41.

[45] M. Dorigo, T. Stutzle. Ant Colony Optimization, MIT Press Cambridge, 2004.

[46] M. Dorigo, L. M. Gambardella, Ant Colony System: A cooperative learning approach to the Traveling Salesman Problem, IEEE Trans. Evolutionary Comput. 1 (1997) 53–66.

[47] J. Dréo, P. A. Siarry, A new Ant Colony algorithm using the heterarchical concept aimed at optimization of multiminima continuous functions, in: M. Dorigo, G. Di Caro, M. Sampels (Eds.), Ant algorithms – Proceedings of ANTS 2002 – Third international workshop, Lecture Notes in Comput. Sci., vol. 2463, Springer Berlin, 2002, pp. 216–221.

[48] P. Du, N. Liu, H. Zhang, J. Lu, An Improved Ant Colony Optimization based on an adaptive heuristic factor for the Traveling Salesman Problem, Hindawi Journal of Advanced Transportation (2021).

[49] S. Ebadinezhad, DEACO: Adopting dynamic evaporation strategy to enhance ACO algorithm for the Traveling Salesman Problem, Eng. Appl. Artif. Intell. 92 (2020) 103649.

[50] W. Elloumi, H. El Abed, A. Abraham, A.M. Alimi, A comparative study of the improvement of performance using a PSO modified by ACO applied to TSP, Appl. Soft. Comput. 25 (2014) 234–241.

[51] A.H. Erol, M. Er, S. Bulkan, Optimizing the Ant Colony Optimization algorithm using neural network for the Traveling Salesman Problem, in: Proceedings of the 2012 International Conference on Industrial Engineering and Operations Management, IEOM, 2012, pp. 1695–1701.

[52] J. B. Escario, J. F. Jimenez, and J. M. Giron–Sierra, Ant Colony extended: Experiments on the Travelling Salesman Problem, Expert Systems with Applications 42 (2015) 390–410.

[53] A. E.-S. Ezugwu, A. O. Adewumi, Discrete Symbiotic Organisms search algorithm for Travelling Salesman Problem, Expert Syst. Appl. 87 (2017) 70–78.

[54] T. Fei, X. Wu, L. Zhang, Y. Zhang, L. Chen, Research on improved Ant Colony Optimization for Traveling Salesman Problem, Mathematical Biosciences and Engineering 19 (2022) 8152–8186.

[55] R. C. Fetecau, A. Guo, A mathematical model for flight guidance in honeybees swarms, Bull. Math. Biol. 74 (2012) 2600–2621.

[56] D. Gaertner, K. L. Clark, On Optimal Parameters for Ant Colony Optimization Algorithms, in: Proc. of the 2005 International Conference on Artificial Intelligence (IC-AI), Las Vegas, Nevada, USA, pp. 83-89, 2005.

[57] L. M. Gambardella, M. Dorigo, Solving symmetric and asymmetric TSPs by ant colonies, in: T. Baeck, T. Fukuda, Z. Michalewicz (Eds.), Proceedings of the 1996 IEEE international conference on evolutionary computation, IEEE Press Piscataway, 1996, pp. 622–627.

[58] L. M. Gambardella, M. Dorigo, Ant Colony System hybridized with a new local search for the sequential ordering problem, Informs. J. Comput. 12(2000) 237–255.

[59] L.M. Gambardella, M. Dorigo, Ant-Q: a reinforcement learning approach to the Travelling Salesman Problem, in: Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann, Palo Alto, California, USA (2000), 252–260.

[60] Y. Gan, L.W. Lan, S. Li, Study on parameters configuration for Ant Colony Optimization, Adv. Mater. Res. 279 (2011) 371–376.

[61] G. H. Al-Gaphari, R. Al-Amry, A. S. Al-Nuzaili, Discrete crow-inspired algorithms for Traveling Salesman Problem, Eng. Appl. Artif. Intell. 97 (2021) 104006.

[62] M. R. Garey, D. S. Johnson, Computers and intractability; A guide to the theory of NP Completeness, WH Freeman New York, 1979.

[63] X.T. Geng, Z.H. Chen, W. Yang, D.Q. Shi, K. Zhao, Solving the Traveling Salesman Problem based on an adaptive simulated annealing algorithm with greedy search, Appl. Soft. Comput. 11 (2011) 3680–3689.

[64] M. L. Ginsberg, Essentials of artificial intelligence, Morgan Kaufmann San Mateo, 1993

[65] F. Glover, Future paths for integer programming and links to artificial intelligence, Comput. Oper. Res. 13 (1986) 533–549.

[66] Handbook of meta-heuristics, F. Glover, G. Kochenberger (Eds.), Kluwer Academic Norwell, 2002.

[67] D. Gomez-Cabrero, D. Ranasinghe, Fine-tuning the Ant Colony System Algorithm through Particle Swarm Optimization, in: Proc. of the International Conference on Information and Automation, 2005.

[68] S. Goss, R. Beckers, J. L. Denebourg, S. Aron, J. M. Pasteels, How trail laying and trail following can solve foraging problems for ant colonies, in: Behavioural Mechanisms of Food Selection, R. N. Hughes (Ed.), NATO ASI Series, vol. G 20, Springer-Verlag Berlin, 1990.

[69] J.J. Grefenstette, R. Gopal, B. Rosmaita, D. Van Gucht, Genetic algorithms for the Traveling Salesman Problem, in: Proceedings of the First International Conference on Genetic Algorithms and their Applications, 1985, pp. 160–168.

[70] S. Gulcu, M. Mahi, O. K. Baykan, H. Kodaz, A parallel cooperative hybrid method based on ant colony optimization and 3-Opt algorithm for solving Traveling Salesman Problem, Soft Comput. 22 (2018) 1669–1685.

[71] S. Gupta, K. Deep, A memory-based grey wolf optimizer for global optimization tasks, Appl. Soft. Comput. 93 (2020) 106367.

[72] I. K. Gupta, S. Shakil, S. Shakil, A hybrid GA-PSO algorithm to solve Traveling Salesman Problem, Computational Intelligence: Theories, Applications and Future Directions – Volume I, vol. 798, pp. 453–462, 2019.

[73] W. J. Gutjahr, A graph-based Ant System and its convergence, Future Generat. Comput. Syst. 16 (2000) 873–888.

[74] W. J. Gutjahr, ACO algorithms with guaranteed convergence to the optimal solution, Informat. Process. Lett. 82 (2002) 145–153.

[75] B. Hajek, Cooling schedules for optimal annealing, Math. Oper. Res. 13 (1988) 311–329.

[76] Z.-F. Hao, R.-C. Cai, and H. Huang, An adaptive parameter control strategy for ACO, in: Proc. of 2006 International Conference on Machine Learning and Cybernetics, Dalian, China, pp. 203–206, 2006.

[77] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, Harris Hawks Optimization: Algorithm and applications, Future Generation Computer Systems 97 (2019) 849–872.

[78] M. L. He, Z. X. Wei, X. H. Wu, Y. T. Peng, An improved Ant Colony Optimization algorithm for vehicle routing problem with soft time windows, Comput. Integr. Manuf. Syst., in press.

[79] M. Held, R.M. Karp, A dynamic programming approach to sequencing problems, J. Soc. Ind. Appl. Math. 10 (1962) 196–210.

[80] H. P. Hipolito, S. G. Juan-Jose, A Branch-and-cut algorithm for the split-demand one-commodity pickup-and-delivery Travelling Salesman Problem, Eur. J. Oper. Res. 297 (2022) 467–483.

[81] K. L. Hoffman, M. Padberg, G. Rinaldi, Encyclopedia of Operations Research and Management Science; Springer: Boston, MA, USA, 2013.

[82] J. H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press Ann Arbor, 1975.

[83] S. Hougardy, F. Zaiser, X.H. Zhong, The approximation ratio of the 2-Opt Heuristic for the metric Traveling Salesman Problem, Oper. Res. Lett. 48 (2020) 401–404.

[84] Y. J. Hu, Z. Zhang, Y. Yao, X. P. Huyan, X. S. Zhou, W. S. Lee, A bidirectional graph neural network for Traveling Salesman Problems on arbitrary symmetric graphs, Eng. Appl. Artif. Intell. 97 (2021) 104061.

[85] Y. Huang, X. N. Shen, X. You, A discrete shuffled frog-leaping algorithm based on heuristic information for Traveling Salesman Problem, Appl. Soft Comput. 102 (2021) 107085.

[86] D.S. Johnson and L.A. McGeoch, The Travelling Salesman Problem: A case study in local optimization, in: E. H. L. Aarts and J. K. Lenstra (Eds.), Local Search in Combinatorial Optimization, pages 215–310. John Wiley & Sons, 1997.

[87] S. K. R. Kanna, K. Sivakumar, N. Lingaraj, Development of Deer Hunting linked Earthworm Optimization Algorithm for solving large scale Traveling Salesman Problem, Knowledge-Based Syst., 227 (2021) 107199.

[88] R.M. Karp, On the computational complexity of combinatorial problems, Networks 5 (1975) 331–333.

[89] O. Karpenko, J. Shi, Y. Dai, Prediction of MHC class II binders using the Ant Colony search strategy, Artificial Intelligence in Medicine 35 (2005) 147–156.

[90] J. Kennedy, R. C. Eberhart, Particle Swarm Optimization, in: Proceedings of the 1995 IEEE international conference on neural networks, vol. 4., IEEE Press Piscataway, 1995, pp. 1942–1948.

[91] I. Khan, M. K. Maiti, A swap sequence based Artificial Bee Colony algorithm for Traveling Salesman Problem, Swarm and Evolutionary Computation 44 (2018) 428–438.

[92] I. Khan, M. K. Maiti, M. Maiti, Coordinating Particle Swarm Optimization, Ant Colony Optimization and K-Opt Algorithm for Traveling Salesman Problem, Commun. Comput. Inf. Sci. 655 (2017) 103–119.

[93] M. M. Krishna, N. Panda, S. K. Majhi, Solving Traveling Salesman Problem using hybridization of Rider Optimization and Spotted Hyena Optimization algorithm, Expert Syst. Appl.183 (2021).

[94] G. Laporte, The Traveling Salesman Problem: An overview of exact and approximate algorithms, Eur. J. Oper. Res. 59 (1992) 231–247.

[95] The Travelling Salesman Problem, E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy-Kan, D. B. Shmoys (Eds.), Wiley New York, 1985.

[96] R. Leardi, Genetic algorithms, Comprehensive Chemometrics 1 (2009) 631–653.

[97] W. Lei, F. Wang, Research on an improved Ant Colony Optimization algorithm for solving Traveling Salesmen Problem, International Journal of Database Theory and Application 9 (2016) 25–36.

[98] J. K. Lenstra, A. H. G. Rinnooy-Kan, Some simple applications of the Travelling Salesman Problem, J. Oper. Res. Soc. 26 (1975) 717–733.

[99] L. Li, Y. Cheng, L. Tan, B. Niu, A discrete Artificial Bee Colony algorithm for TSP problem, Int. J. Adv. Comput. Technol. 4 (2012) 566–573.

[100] W. H. Li,W Y. Yang, H. Q. Liao, J. L. Li, X. P. Zheng, Artificial Bee Colony algorithm for Traveling Salesman Problem, Adv. Mater. Res. 314 (2011) 2191–2196.

[101] J. Li, Y. Xia, B. Li, Z. G. Zeng, A pseudo-dynamic search Ant Colony Optimization algorithm with improved negative feedback mechanism, Cognit. Syst. Res. 62 (2020) 1–9.

[102] M. L. Li, Q. Z. Li, Path planning of unmanned crane based on improved Ant Colony Algorithm, Comput. Simul. 38 (2021) 172–176+226.

[103] E. Liao, C. A. Liu, A hierarchical algorithm based on density peaks clustering and Ant Colony Optimization for Traveling Salesman Problem, IEEE Access 6 (2018) 38921–38933.

[104] S. Lin, Computer solutions for the Traveling Salesman Problem, Bell Systems Technology Journal 44 (1965) 2245–2269.

[105] S. Lin, B. W. Kernighan, An effective Heuristic Algorithm for the TSP, Operations Research 21 (1973) 498 – 516.

[106] M. Lindauer, Schwarmbienen auf wohnungssuche, Zeitschrift für Vergleichende Physiologie 37 (1955) 263–324.

[107] C. Liu, Y.K. Du, A membrane algorithm based on chemical reaction optimization for many-objective optimization problems, Knowl.-Based Syst. 165 (2019) 306–320.

[108] Y. Liu, Research on the algorithm optimization of improved ant colony algorithm-LSACA, International Journal of Signal Processing, Image Processing and Pattern Recognition 9 (2016) 143–154.

[109] C. Liu, L. Wu, X. D. Huang, W. S. Xiao, Improved dynamic adaptive Ant Colony Optimization algorithm to solve pipe routing design, Knowledge-Based Syst., 237 (2022) 107846.

[110] M. Liu, Y. Li, A. Li, Q. Huo, N. Zhang, N. Qu, M. Zhu, L. Chen, A Slime Mold-Ant Colony fusion algorithm for solving Traveling Salesman Problem, IEEE Access 8 (2020) 202508..202521.

[111] Y. Liu, B. Cao, A Novel Ant Colony Optimization Algorithm with Levy Flight, IEEE Access 8 (2020) 67205–67213.

[112] E. Lumer, B. Faieta, Diversity and adaptation in populations of clustering ants, in: J.-A. Meyer, S. W. Wilson (Eds.), Proceedings of the third international conference on simulation of adaptive behavior: From animals to animats, vol. 3, MIT Press Cambridge, 1994, pp. 501 – 508.

[113] M. Mahi, O. K. Baykan, H. Kodaz, A new hybrid method based on Particle Swarm Optimization, Ant Colony Optimization and 3-Opt algorithms for Traveling Salesman Problem, Applied Soft Computing 30 (2015) 484–490.

[114] K. Marriott, P. Stuckey, Programming with constraints, MIT Press Cambridge, 1998.

[115] R. Masadeh, A. Sharieh, B.A. Mahafzah, Humpback whale optimization algorithm based on vocal behavior for task scheduling in cloud computing, Int. J. Adv. Sci. Tech. 13 (2019) 121–140.

[116] M. Mavrovouniotis, F. M. Muller, S. Yang, Ant colony optimization with local search for dynamic Traveling Salesman Problems, IEEE. Trans. Cybern. 47 (2017) 1743–1756.

[117] M. Mavrovouniotis, S. Yang, Ant Colony Optimization with immigrants schemes for the dynamic Travelling Salesman Problem with traffic factors, Appl. Soft Comput. 13 (2013) 4023–4037.

[118] L. Melo, F. Pereira, E. Costa, MC-ANT: A Multi-Colony Ant Algorithm, in: Proc. of International Conference on Artificial Evolution (Evolution Artificielle), Strasbourg, France, pp. 25–36, 2009.

[119] D. Merkle, M. Middendorf, H. Schmeck, Ant Colony Optimization for resource–constrained project scheduling. IEEE Trans. Evolutionary Comput. 6 (2002) 333–346.

[120] S. Mirjalili, Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, Neural Computing and Applications, 27 (2016) 1053–1073.

[121] J. D. Moss, C. G. Johnson, An ant colony algorithm for multiple sequence alignment in bioinformatics, in: D. W. Pearson, N. C. Steele, R. F. Albrecht (Eds.), Artificial neural networks and genetic algorithms, Springer Berlin, 2003, pp. 182–186.

[122] N. Monmarché, G. Venturini, M. Slimane, On how pachycondyla apicalis ants suggest a new search algorithm, Future Generation Comput. Syst. 16 (2000) 937–946.

[123] J. J. Muren, L. Wu, Z. P. Zhou, Y. L. Du, Mixed steepest descent algorithm for the Traveling Salesman Problem and application in air logistics, Transp. Res. E 126 (2019) 87–102.

[124] G. L. Nemhauser, A. L. Wolsey, Integer and combinatorial optimization, John Wiley & Sons New York, 1988.

[125] E. Osaba, J.D. Ser, A. Sadollah, M.N. Bilbao, D. Camacho, A discrete water cycle algorithm for solving the symmetric and asymmetric Traveling Salesman Problem, Appl. Soft. Comput. 71 (2018) 277–290.

[126] A. Ouaarab, B. Ahiod, X.-S. Yang, Discrete cuckoo search algorithm for the Travelling Salesman Problem, Neural Comput. Appl. 24 (2013) 1659–1669.

[127] P. S. Ow, T. E. Morton, Filtered beam search in scheduling, Internat. J. Production Res. 26 (1988) 297–307.

[128] A. Panda, S. Pani, A Symbiotic Organisms Search algorithm with adaptive penalty function to solve multi-objective constrained optimization problems, Appl. Soft. Comput. 46 (2016) 344–360.

[129] K. Panwar, K. Deep, Discrete Grey Wolf Optimizer for symmetric Travelling Salesman Problem, Appl. Soft Comput. 105 (2021) 107298.

[130] C. H. Papadimitriou, K. Steiglitz, Combinatorial optimization – Algorithms and complexity, Dover New York, 1982.

[131] M. Peker, B. Sen, P.Y. Kumru, An efficient solving of the Traveling Salesman Problem: the Ant Colony System having parameters optimized by the Taguchi method, Turk. J. Electr. Eng. Comput. 21 (2013) 2015–2036.

[132] C.Peterson, Parallel distributed approaches to combinatorial optimization: Benchmark studies on Traveling Salesman Problem, Neural Computation 2 (1990) 261–269.

[133] Z. J. Peya, M. A. H. Akhand, T. Sultana, M. M. H. Rahman, Distance based Sweep Nearest Algorithm to solve capacitated vehicle routing problem, Int. J. Adv. Comput. Sci. Appl. 10 (2019) 259–264.

[134] M. L. Pilat, T. White, Using Genetic Algorithms to Optimize ACS-TSP, In: Proc. of International workshop on ant algorithms, Brussels, Belgium, pp. 282–287, 2002.

[135] M. O. R. Prates, P. H. C. Avelar, H. Lemos, L. Lamb, M. Vardi, Learning to solve np-complete problems: A graph neural network for decision TSP, in: Proceedings of the AAAI Conference on Artificial Intelligence 33 (2019) 4731–4738.

[136] A. P. Punnen, The Traveling Salesman Problem: Applications, Formulations and Variations. In The Traveling Salesman Problem and Its Variations; Springer: Boston, MA, USA, 2007; Volume 12, pp. 1–28.

[137] A. Ragmani, A. Elomri, N. Abghour, K. Moussaid, M. Rida, An improved hybrid fuzzy-ant colony algorithm applied to load balancing in cloud computing environment, Procedia Comput. Sci. 151 (2019) 519–526.

[138] S. S. Rajesh Matai, M. L. Mittal, Traveling Salesman Problem: An overview of applications, formulations, and solution approaches, in: Tech, West Palm Beach, FL, USA, 2011.

[139] G. Reinelt, The Traveling Salesman: Computational solutions for TSP Applications, volume 840 of LNCS. Springer Verlag, 1994.

[140] F. Romeo, A. Sangiovanni–Vincentelli, A theoretical framework for simulated annealing, Algorithmica 6 (1991) 302 – 345.

[141] S. K. Sahana, An automated parameter tuning method for Ant Colony Optimization for scheduling jobs in grid environment, International Journal of Intelligent Systems and Applications 11 (2019) 11.

[142] B. P. Sahoo, S. Panda, Load frequency control of solar photovoltaic/wind/biogas/biodiesel generator based isolated microgrid using Harris Hawks Optimization, in: Proc. of the 2020 First International Conference on Power, Control and Computing Technologies (ICPC2T), Raipur, India, pp. 188–193, 2020.

[143] Y. Saji, M. Barkatou, A discrete bat algorithm based on Levy flights for Euclidean Traveling Salesman Problem, Expert Syst. Appl. 172 (2021) 114639.

[144] H. Savuran, M. Karakaya, Efficient route planning for an unmanned air vehicle deployed on a moving carrier, Soft. Comput. 20 (2016) 2905–2920.

[145] K. M. Schultz, K. M. Passino, T. D. Seeley, The mechanism of flight guidance in honeybee swarms: subtle guides or streaker bees?, Journal of Experimental Biology 211 (2008) 3287–3295.

[146] T. D. Seeley, Honeybee democracy, Princeton University Press Princeton, 2010.

[147] T. D. Seeley, S. C. Buhrman, Group decision making in swarms of honey bees, Behav. Ecol. Sociobiol. 45 (1999), 19–31.

[148] A. S. B. Shahadat, M. A. H. Akhand, M. A. S. Kamal, Visibility adaptation in Ant Colony Optimization for solving Traveling Salesman Problem, Mathematics (2022).

[149] H. Sharma, G. Hazrati, J. C. Bansal, Spider Monkey Optimization algorithm, in: Studies in Computational Intelligence; Springer International Publishing: Berlin/Heidelberg, Germany 779 (2019) 43–59.

[150] A. Shmygelska, R. Aguirre-Hernández, H. H. Hoos, An Ant Colony Optimization algorithm for the 2D HP protein folding problem, in: M. Dorigo, G. Di Caro, M. Sampels (Eds.), Ant algorithms – Proceedings of ANTS 2002 –Third international workshop, Lecture Notes in Comput. Sci., vol. 2463, Springer Berlin, 2002, pp. 40–52.

[151] A. Shmygelska, H. H. Hoos, An Ant Colony Optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem, BMC Bioinformatics 6 (2005) 1– 22.

[152] C. S. Siang, W. Li-Pei, L. C. Peng, An Artificial Bee Colony algorithm with a modified choice function for the Traveling Salesman Problem, Swarm and Evolutionary Computation 44 (2019) 622–635.

[153] K. Sorensen, Metaheuristics - the metaphor exposed, International Transactions in Operational Research 22 (2015) 3 – 18.

[154] K. Sorensen, M. Sevaux, F. Glover, A history of metaheuristics, in Handbook of Heuristics, pp. 791–808, Springer, Cham, Berlin, 2018.

[155] T. Stutzle, Local Search Algorithms for combinatorial problems: Analysis, improvements, and new applications. PhD thesis, Darmstadt University of Technology, Department of Computer Science, 1998.

[156] T. Stutzle, H. H. Hoos, $\mathcal{MAX} - \mathcal{MIN}$ Ant system, Future Generat. Comput. Syst. 16 (2000) 889–914.

[157] T. Stutzle, M. Dorigo, A short convergence proof for a class of ACO algorithms, IEEE Trans. Evolutionary Comput. 6 (2002) 358–65.

[158] T. Stutzle, H. H. Hoos, The $\mathcal{MAX} - \mathcal{MIN}$ Ant System and Local Search for the Traveling Salesman Problem, in: T. Baeck, Z. Michalewicz, X. Yao (Eds.) Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC 97), 309–314, 1997.

[159] T. Stutzle, H. H. Hoos, Improvements on the Ant System: Introducing the $\mathcal{MAX} - \mathcal{MIN}$ Ant System, in: R. F. Albrecht, G. D. Smith, N. C. Steele (Eds.) Artificial Neural Networks and Genetic Algorithms, pages 245–249. Springer Verlag, Wien New York, 1998.

[160] T. Stutzle, H. H. Hoos. $\mathcal{MAX} - \mathcal{MIN}$ Ant System and Local Search for Combinatorial Optimization Problem, in: S. Voss, S. Martello, I.H. Osman, C. Roucairol (Eds.) Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, pages 313–329. Kluwer, Boston, 1999.

[161] C. Sudipta, M. Mohammad, T. Huseyin, L. Bian, B. William, A modified Ant Colony Optimization algorithm to solve a dynamic Traveling Salesman Problem: A case study with drones for wildlife surveillance, Journal of Computational Design and Engineering 6 (2019) 368–386.

[162] A. F. Tuani, E. Keedwell, M. Collett, Heterogenous adaptive Ant Colony Optimization with 3-opt local search for the Travelling Salesman Problem, Appl. Soft Comput. 97 (2020) 106720.

[163] Y. Wang, J. B. Remmel, A binomial distribution model for the Traveling Salesman Problem based on frequency quadrilaterals, J. Graph. Algorithms Appl. 20 (2016) 411–434.

[164] J. Wang, O. K. Ersoy, M. He, F. Wang, Multi-offspring genetic algorithm and its application to the Traveling Salesman Problem, Applied Soft Computing 43 (2016) 415–423.

[165] Y. Wang, The hybrid genetic algorithm with two local optimization strategies for Traveling Salesman Problem, Comput. Ind. Eng. 70 (4) (2014) 124–133.

[166] L. W. Yang, L. X. Fu, N. Guo, Z. Yang, H. Q. Guo, X. Y. Xu, Path planning with multifactor improved Ant Colony algorithm, Comput. Integr. Manuf. Syst., in press.

[167] S. B. Wang, R. Hu, B. Qian, M. Y. Liu, Improved Ant Colony Optimization for solving green periodic vehicle routing problem, Control Eng. China, in press.

[168] Y. Wang, Y.W. Wu, N. Xu, Discrete Symbiotic Organism search with excellence coefficients and self-escape for Traveling Salesman Problem, Comput. Ind. Eng. 131 (2019) 269–281.

[169] Y. Wang, Z. Han, Ant Colony Optimization for Traveling Salesman Problem based on parameters optimization, Applied Soft Computing 107 (2021) 107439.

[170] C.J.C.H. Watkins, P. Dayan. Q-Learning. Machine Learning 8 (1992) 279–292.

[171] D. Whitley, T. Starkweather, D. Fuquay, Scheduling problems and travelling salesmen: The genetic edge recombination operator, Proc. of the Third Int. Conf. on Genetic Algorithms, Morgan Kaufmann, 1989

[172] H. Xu, X. Qian, L. Zhang, Study of ACO algorithm optimization based on improved tent chaotic mapping, Journal of Information and Computational Science 9 (2012) 1653–1660.

[173] Y. Yan, H.-S. Sohn, G. Reyes, A modified Ant System to achieve better balance between intensification and diversification for the Traveling Salesman Problem, Applied Soft Computing 60 (2017) 256–267.

[174] L. W. Yang, L. X. Fu, N. Guo, Z. Yang, H. Q. Guo, X. Y. Xu, Path planning with multifactor improved ant colony algorithm, Comput. Integr. Manuf. Syst., in press.

[175] M. Yazdani, F. Jolai, Lion Optimization Algorithm (LOA): A nature-inspired metaheuristic algorithm, J. Comput. Des. Eng. 3 (2015) 24–36.

[176] S. A. Yasear, K. R. Ku-Mahamud, Fine-tuning the Ant Colony System algorithm through Harris's Hawk Optimizer for Travelling Salesman Problem, International Journal of Intelligent Engineering and Systems.

[177] J. Zhang, L. Hong, Q. Liu, An improved whale optimization algorithm for the Traveling Salesman Problem, Symmetry 13 (2020) 48.

[178] J. Y. Zheng, X. Q. Cheng, J. J. Fu, Application research of improved ant colony algorithm in TSP, Comput. Simul 38 (2021) 126–130.

[179] Y. W. Zhong, J. Lin, L. J. Wang, H. Zhang, Discrete comprehensive learning Particle Swarm Optimization algorithm with Metropolis acceptance criterion for Traveling Salesman Problem, Swarm. Evol. Comput. 42 (2018) 77–88.