Mathematical Runtime Analysis of ACO Algorithms: Survey on an Emerging Issue

Walter J. Gutjahr
Department of Statistics and Decision Support Systems
University of Vienna

Abstract: The paper gives an overview on the status of the theoretical analysis of Ant Colony Optimization (ACO) algorithms, with a special focus on the analytical investigation of the runtime required to find an optimal solution to a given combinatorial optimization problem. First, a general framework for studying questions of this type is presented, and three important ACO variants are recalled within this framework. Secondly, two classes of formal techniques for runtime investigations of the considered type are outlined. Finally, some available runtime complexity results for ACO variants, referring to elementary test problems that have been introduced in the theoretical literature on evolutionary algorithms, are cited and discussed.

1 Introduction

A metaheuristic that belongs now to the most prominent and most frequently applied techniques for search and heuristic optimization started its development fifteen years ago out of the seminal work by Dorigo, Maniezzo and Colorni [11], [12]: the ant-based approach to the solution of combinatorial optimization problems. Originally motivated by the attempt to solve the well-known Travelling Salesperson Problem (TSP), the inventors of the approach recognized soon that their technique is applicable to a much larger range of problems. In an explicit form, this insight was established by the creation of the *Ant Colony Optimization* (ACO) metaheuristic by Dorigo and Di Caro in [9]. In the meantime, there exist several hundreds of publications reporting on successful applications of ACO in a large variety of areas. For a recent survey, we refer the reader to the profound and comprehensive textbook by Dorigo and Stützle [13].

Notwithstanding the mentioned fact that there are already numerous experimental investigations on ACO, much less has been done in the field of ACO *theory*. It was not before the year 2000 that the first *convergence proofs* for ACO algorithms appeared (see [17], [30], [18]). At present, convergence results are already available for several ACO variants. Even if an ACO algorithm is ensured to converge to an optimal solution, there remains the practically relevant question of how much time it takes (e.g., in the average) until such a solution is found. Contrary to the situation in the Genetic Algorithms (GA) field, results concerning this last question are still scarce in the ACO area.

As soon as one passes from the question of convergence of a metaheuristic algorithm to that of the *speed* of convergence, one cannot expect anymore to obtain very *general* (positive) results. This is a consequence of the so-called "no-free-lunch theorems" which basically state that for

each algorithm that performs well on *some* optimization problems, there must necessarily be *other* problems where it fails to be efficient. As a consequence, runtime analysis of a metaheuristic algorithm has to be done on a more detailed level by investigating various specific test problems. These test problems may seem unrealistically simple from the viewpoint of applied research, but studying them can provide helpful information in so far as each of these problems incorporates some typical feature (or several features) of real problems. Hence, their separate analysis helps us to *understand* why and when certain algorithmic variants or parametrizations work resp. do not work in real-life applications.

Although the analytical investigation of the runtime of ACO algorithms is still a very new issue at the moment, there are already some first results available, and they promise that a large amount of knowledge on this issue can be won within the next years. Some of the existing results already allow a comparison of specific ACO algorithms with counterparts from the field of Evolutionary Algorithms, in particular with the so-called (1+1)-EA (see [14]).

The aim of the present survey article is to describe a general, well-defined formal framework based on which results of the outlined type can be derived in a mathematically sound way, to present already available results, to give an introduction into the techniques by which these results have been obtained, to outline their scope of application, and to discuss topics for future research.

The paper is organized as follows: Section 2 presents some important ACO variants within a general framework, that of the construction graph in the sense of [17], [18], which lends itself very well for theoretical investigations on ACO. Section 3 deals with the following questions: Under which conditions and in which sense can convergence to optimal solutions be ensured? Section 4 presents general-purpose techniques for analytical investigations of ACO algorithms. In section 5, available results concerning two important ACO variants (Ant System and $\mathcal{MM}$AS) are outlined and discussed. Section 6 gives concluding remarks.

2 A Short Recapitulation of Some ACO Variants

2.1 Construction Graphs

Before specifying some particular ACO algorithms, we start with the definition of a unifying ACO framework procedure based on the *construction graph* concept in the form as developed in [17], [18].[1] Consider a combinatorial optimization problem of the form

$$f(x) \to \max, \quad x \in S. \tag{1}$$

Therein, x is a *solution* to the combinatorial optimization problem, S is a finite set of feasible solutions (the so-called *feasible set*), and f is the *objective function*, also called *fitness function*. To have a natural interpretation of "fitness", we formulate all problems as maximization problems. The combinatorial optimization problem can be constrained or unconstrained; for (1),

[1] The construction graph definition used here is different from that given in [13], [8], the main difference being that we identify solution components with the *edges* of the graph, whereas in [13], [8], solution components are identified with its *nodes*. Both types of construction graphs have their respective advantages. We choose here the type described in [17], [18] because it allows very natural representations, giving visual intuition, for the standard problems treated by ACO. E.g., in the case of a TSP, our construction graph has the same structure as the underlying transportation graph, the n nodes representing the n cities (cf. Fig. 1).

this distinction is not relevant, provided that S is considered as the set of solutions satisfying all given constraints.

Let an instance of a combinatorial optimization problem of the form (1) be given. By a *construction graph* for this instance, we understand a directed graph $\mathcal{C} = (\mathcal{V}, \mathcal{A})$ with node set $\mathcal{V}$ and edge set $\mathcal{A}$, together with a *solution decoding function* Φ, with the following properties:

(1) In $\mathcal{C}$, a unique node is marked as the so-called *start node*.

(2) Let $\mathcal{W}$ be the set of directed paths w in $\mathcal{C}$, called *feasible paths*, satisfying the following conditions:

   (a) w starts at the start node of $\mathcal{C}$;

   (b) w contains each node of $\mathcal{C}$ at most once;

   (c) w is maximal in $\mathcal{W}$, that is, it cannot be prolonged (by appending edges and nodes) to a longer feasible path in $\mathcal{W}$.

   The function Φ maps the set $\mathcal{W}$ of feasible paths onto the set of feasible solutions S of the given problem instance: To each feasible path $w \in \mathcal{W}$, there corresponds a feasible solution $x = \Phi(w) \in S$, and to each feasible solution $x \in S$, there corresponds at least one feasible path in $\mathcal{W}$ such that $\Phi(w) = x$.

Example 1. The graph in Fig. 1 is the natural construction graph for an *asymmetric TSP* with $n = 5$ cities. Node 1 is the start node. Here, $\mathcal{C}$ is a complete graph, and a feasible path is simply a path starting in node 1 and containing each node ("city") exactly once. The decoding function Φ takes such a feasible path w and assigns to it a feasible solution (i.e., a closed tour) of the TSP by adding an additional move from the last node of w back to the start node, such that the tour becomes closed.—If one prefers to leave the choice of the first node of the tour open, as it is done in many ACO applications to the TSP, one may use an alternative construction graph obtained by adding an artificial start node 0 and directed edges leading from this node to each of the nodes $1, \ldots, n$.

Example 2. The graph in Fig. 2 (called *chain* in [20]) shows a possible construction graph for a *subset selection problem*, the feasible solutions of which are the subsets of a set $\{1, \ldots, n\}$ of items. To describe a solution, we let component x_i of the solution vector x take the value 1 if item i is to be included in the subset, and the value 0 otherwise. The fitness function $f : \{0,1\}^n \to R$ can be an arbitrary function. The *chain* construction graph encodes the choice $x_i = 1$ as an up-move from node $\underline{i-1}$ to node i, and the choice $x_i = 0$ as a down-move from node $\underline{i-1}$ to node $-i$ $(i = 1, \ldots, n)$. The moves from a node i or $-i$ to their successor node $\underline{i}$ are deterministic and do not play a role for the decision process. Node $\underline{0}$ is the start node. As soon as node $\underline{n}$ is reached, the walk terminates. As in the TSP construction graph of Fig. 1, there are also here no special restrictions on feasible continuations except those implicitly given by the graph structure, such that the set $\mathcal{W}$ of feasible paths is just the set of directed paths of length $2n$ in the chain graph.

Example 3. Another possible construction graph (called *drum* in [20]) for a subset selection problem is depicted in Fig. 3. It encodes the solutions in a more redundant way. Node $\underline{0}$ is the start node. Again, the construction graph contains for each item i two different nodes i and $-i$, but they are now completely interlinked, with the exception of links between nodes i and

$-i$ and from node i or $-i$ to node $\underline{0}$. For each item, the decision on whether to set $x_i = 1$ or $x_i = 0$ is represented by the decision whether node i or node $-i$, respectively, is visited by the path. Obviously, in this encoding, the same solution x is represented by several different paths. E.g., path $(\underline{0}, -2, 4, 3, 1)$ and path $(\underline{0}, 4, -2, 1, 3)$ are decoded as the same solution $(1, 0, 1, 1)$. Evidently, we have here to impose an additional restriction on paths to make them encodings of feasible solutions: Whenever a node i or $-i$ has been visited, neither i nor $-i$ are allowed to be visited in a future move. The set $\mathcal{W}$ of feasible paths is the set of all directed paths of maximum length in the drum graph respecting this constraint. The different consequences of different construction graph choices for the same problem instance will become clear after the presentation of the ACO algorithm in the following subsection.

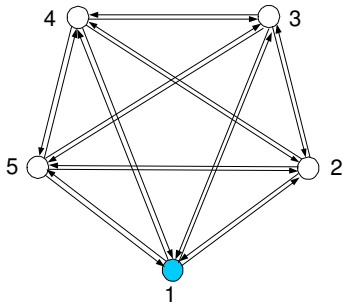

Fig. 1. Construction graph for an asymmetric TSP with $n = 5$ cities.

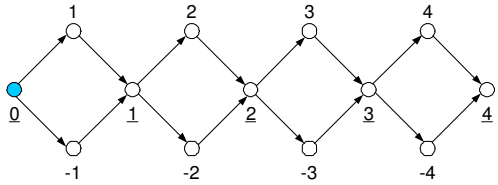

Fig. 2. "Chain" construction graph for a subset selection problem with $n = 4$.

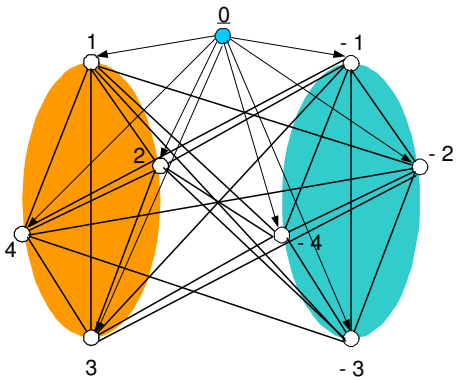

Fig. 3. "Drum" construction graph for a subset selection problem with $n = 4$. Undirected edges represent pairs of directed edges with opposite orientation.

The *objective function value* (fitness value) assigned to a feasible path w is always that of $\Phi(w)$, that is, that of the corresponding feasible solution x.

By a *partial path* in $\mathcal{C}$, we understand a path u that satisfies properties (2a) – (2b) in the construction graph definition, but not necessarily property (2c). A *continuation* of a partial

path u with node i as its last node is an edge $(i, j) \in \mathcal{A}$. A continuation (i, j) of u is called *feasible* if a feasible path $w \in \mathcal{W}$ starting with u and traversing then edge (i, j) exists, otherwise it is called *infeasible*. E.g., the continuation $(3, 4)$ of $u = (\underline{0}, -2, 3)$ in the *drum* graph of Fig. 3 is feasible, whereas the continuation $(3, 2)$ of u is infeasible, because once node -2 has been visited, node 2 is forbidden in all subsequent moves by our additional restriction indicated above.

Let us give a few explanations to the elements of the construction graph definition above, indicating that they are all necessary to obtain meaningful results: Condition (1) together with condition (2a) is required to define where the solution construction (in our framework represented by the traversal of a path) starts. Condition (2b) avoids cycling of the procedure. Moreover, it turns out to be an indispensable precondition for the validity of certain convergence results. Condition (2c) ensures that yet incomplete partial paths can be clearly distinguished from complete paths. This is necessary in order that the algorithm "knows" when to start the decoding of a path to a solution by applying function Φ.

2.2 Framework Algorithm

Based on our construction graph definition, a general framework algorithm encompassing different well-known versions of ACO is given by the procedure described below.

Procedure ACO

Initialize pheromone trails τ_{ij} on the edges (i, j) of $\mathcal{C}$;
for iteration $m = 1, 2, \ldots$ **do**
  for agent $s = 1, \ldots, S$ **do**
    set i, the current position of the agent, equal to the start node of $\mathcal{C}$;
    set u, the current partial path of the agent, equal to the empty list;
    while a feasible continuation (i, j) of the path u of the agent exists **do**
      select successor node j with probability p_{ij}, where
        $p_{ij} = 0$, if continuation (i, j) is infeasible, and
        $p_{ij} = g(\tau_{ij}, \eta_{ij}(u)) \,/\, (\sum_{(i,r)} g(\tau_{ir}, \eta_{ir}(u))$,
          where the sum is over all feasible continuations (i, r), otherwise;
      continue the current path u of the agent by adding edge (i, j) to u
        and setting $i = j$;
    end while
    update the pheromone trails τ_{ij};
  end for
end for

Explanations

- The *pheromone trails* are nonnegative real values assigned to the edges of the construction graph. They serve as the "memory" of the procedure, storing information on how good the single solution components (corresponding to the edges of the construction graph) have turned out in previous iterations. Most ACO variants initialize the pheromone trails by giving them an equal value on all edges.

- The values $\eta_{ij}(u)$ are called *heuristic information values* in the literature. They are obtained by means of some problem-specific heuristic for the combinatorial optimization problem under consideration. It is also possible to work without using heuristic information values. (We shall do that in the next sections.) Contrary to τ_{ij}, the quantities $\eta_{ij}(u)$ are allowed to depend on the entire partial path u traversed, as indicated by the argument u.

- The function g combines pheromone trail and heuristic information. The most popular choice is
$$g(\tau, \eta) = \tau^\alpha \cdot \eta^\beta \tag{2}$$
with parameters $\alpha > 0$ (often chosen as $\alpha = 1$) and $\beta > 0$.

- The diverse ACO variants mainly differ by the way the update of the pheromone trails is performed. This issue is addressed in the following subsection.

2.3 Some Important ACO Variants

We specify now three of the best-known and most frequently applied ACO algorithms within the framework of the last subsection. It has to be mentioned that these specifications only represent the "ant-based" cores of the respective algorithms and do not include additional features such as local post-optimization. Although such extensions, often referred as "daemon actions", are extremely valuable to increase performance in applications, a theoretical runtime analysis has to start with the pure algorithmic variants, where additional procedures are not yet applied.

Ant System (AS). This is the first ACO algorithm that has been proposed in the literature (see Dorigo, Maniezzo and Colorni [11], [12]). It is characterized by the following pheromone update rule:
$$\tau_{ij} := (1 - \rho) \cdot \tau_{ij} + \frac{\rho}{S} \cdot \sum_{s=1}^{S} \Delta\tau_{ij}^s, \tag{3}$$
where $\rho \in [0, 1]$ is the so-called *evaporation rate*, and, denoting by w^s the path of agent s,
$$\Delta\tau_{ij}^s = \begin{cases} C \cdot f(w^s), & \text{if } (i,j) \in w^s, \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$
with a constant $C > 0$. [2] We see that in Ant System, *all* agents contribute to the pheromone increment, the increment itself being chosen as fitness-proportional.

$\mathcal{MAX} - \mathcal{MIN}$ Ant System ($\mathcal{MM}$AS). This approach has been developed by Stützle and Hoos in [31], [32]. It is characterized mainly by two innovations compared to Ant System: First, instead of allowing all agents to deposit pheromone on their paths, only paths that have

[2] In the original publications on AS, the factor ρ/S before the summand in (3) does not appear; in more recent presentations, ρ appears (which makes sense as the reward in the second term of (3) must in some way compensate the loss proportional to ρ in the first term), but not $1/S$. Extracting in (3) not only ρ, but also $1/S$ from the proportionality factor between pheromone increment and fitness makes no essential difference in the definition above (one can imagine that C is multiplied by S for compensation), but will turn out to be useful in subsection 4.2.

turned out as particularly good are reinforced. Two selection rules are used either alternatively or in a combined way. The first is *best-so-far* (BS), where the best path found up to now (no matter in which iteration it occurred) is reinforced, the other is *iteration-best* (IB), where the best path found in the current iteration is reinforced. Secondly, to compensate for the more unbalanced pheromone trail aggregation caused by the restriction to reinforcement only on "outstanding" paths, $\mathcal{MM}$AS applies lower and upper bounds for the pheromone trails. (There are two additional modifications compared to AS suggested in [31] – [32], concerning a special way of pheromone initialization and possible re-initialization of pheromone trails when stagnation occurs. Since these two points seem to be less central for $\mathcal{MM}$AS, they will be omitted here.)

For a fixed considered iteration m, let $\hat{w}$ denote the best path found in one of the iterations, and let $\tilde{w}$ denote the best path found in iteration m itself $(1, \ldots, m)$. The best-so-far path $\hat{w}$ is updated each time a strictly better path than the current $\hat{w}$ is found. Then the pheromone update rule of $\mathcal{MM}$AS is given by

$$\tau_{ij} := \left[(1-\rho)\cdot\tau_{ij} + \rho\cdot\Delta\tau_{ij}^{best}\right]_{\tau_{min}}^{\tau_{max}}, \tag{5}$$

where τ_{min} and τ_{max} with $0 \leq \tau_{min} < \tau_{max}$ are the lower resp. upper pheromone bound, $\tau \mapsto [\tau]_a^b$ denotes the function

$$[\tau]_a^b = \begin{cases} a, & \text{if } \tau < a, \\ \tau, & \text{if } a \leq \tau \leq b, \\ b, & \text{if } \tau > b, \end{cases} \tag{6}$$

and $\Delta\tau_{ij}^{best}$ is given by

$$\Delta\tau_{ij}^{best} = \begin{cases} C\cdot f(w^{best}), & \text{if } (i,j) \in w^{best}, \\ 0, & \text{otherwise}, \end{cases} \tag{7}$$

with $w^{best} = \hat{w}$ and $w^{best} = \tilde{w}$ in the case of best-so-far selection and in the case of iteration-best selection, respectively. We shall refer to the two cases by the abbreviations $\mathcal{MM}$AS$_{bs}$ resp. $\mathcal{MM}$AS$_{ib}$. In this paper, we will not deal with the combined case.

For allowing the theoretical investigations to start with a still simpler situation, we will also consider the case where the reward for the edges on the reinforced paths is not chosen fitness-proportional, but constant, such that formula (7) is replaced by

$$\Delta\tau_{ij}^{best} = \begin{cases} C, & \text{if } (i,j) \in w^{best}, \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

The corresponding variants will be denoted by $\mathcal{MM}$AS$_{bs,co}$ resp. $\mathcal{MM}$AS$_{ib,co}$.

Rank-Based Ant System (AS$_{rank}$). This variant has been developed by Bullnheimer et al. [6]. It is characterized by the idea that agents are sorted in each iteration according to the fitness rank of the paths generated by them. The $R-1$ best agents are allowed then to deposit pheromone on their paths in a scheme with decreasing weights from rank 1 to rank $R-1$. (Additionally, AS$_{rank}$ proposes also to reinforce the best-so-far solution; again, we omit this feature in order to concentrate on the "pure" version of the algorithmic variant.) This gives the following pheromone update rule:

$$\tau_{ij} := (1-\rho)\cdot\tau_{ij} + \rho\cdot\sum_{r=1}^{R-1}(R-r)\,\Delta\tau_{ij}^{[r]}, \tag{9}$$

where $w^{[r]}$ is the rth-best path found in the current iteration (ties are solved randomly), and $\Delta\tau_{ij}^{[r]}$ is defined analogously as in (4). Obviously, for $R = 2$, AS_{rank} becomes identical to $\mathcal{MM}\text{AS}_{ib}$ with $\tau_{min} = 0$ and $\tau_{max} = \infty$.

We do not claim that *all* successful ACO variants fit into the framework of subsection 2.2; for example, *Ant Colony System* (ACS) [10] cannot be expressed within this framework.

3 Convergence

The question on which conditions and in which sense convergence to optimal solutions takes place is not within the focus of the present survey, but can evidently not be ignored whenever the further-reaching question of convergence *speed* arises. Several notions of convergence of a probabilistic heuristic can be distinguished. Below, we define three different notions of convergence of an ACO algorithm, borrowing terminology from [13]. In the definitions, we use $\tau(m) = (\tau_{ij}(m))_{(i,j)\in\mathcal{A}}$ to denote the vector of pheromone trails in iteration m. The expression $\hat{w}(m)$ denotes the best-so-far path in iteration m. By $w(\tau)$, we denote the (random) path of an agent given that in the current iteration the pheromone trails are described by pheromone vector $\tau = (\tau_{ij})_{(i,j)\in\mathcal{A}}$. By $\mathcal{W}^*$, we denote the set of optimal paths with respect to the given fitness function f. The optimal (i.e., maximal) fitness value will be written as f^*. Finally, the probability of an event A will be denoted by $Pr(A)$.

We start by formal definitions of some convergence notions for ACO.

- *Convergence in solution* holds, if

$$Pr\{w(\tau(m)) \in \mathcal{W}^*\} \to 1 \quad (m \to \infty). \tag{10}$$

  This property is satisfied if the pheromone vector evolves in such a way that it tends more and more to support (only) the generation of an optimal solution. In other words, convergence in solution means that the agents *learn* to produce the optimal solution. This is the strongest notion of convergence we investigate here.

- *ϵ-convergence in solution* for some value $\epsilon \in]0, 1[$ holds, if there exists an iteration number $M(\epsilon)$ such that

$$Pr\{w(\tau(m)) \in \mathcal{W}^*\} \geq 1 - \epsilon \quad \text{for all } m \geq M(\epsilon). \tag{11}$$

  ϵ-convergence in solution is a weaker form of convergence in solution which does not require that the probabilities of producing an optimal solution tend to one, but only that their distance to one becomes smaller than some pre-defined threshold value ϵ. A certain parametrization of an ACO variant can, for example, possess ϵ-convergence in solution for $\epsilon = 0.001$, but not for $\epsilon = 0.0001$. If, for some *fixed* parametrization of the algorithm, the threshold value ϵ can be chosen arbitrarily and for each choice, (11) is satisfied for sufficiently large $M(\epsilon)$, we have the stronger property of convergence in solution.

- *Convergence in value* holds, if

$$Pr\{f(\hat{w}(m)) = f^*\} \to 1 \quad (m \to \infty). \tag{12}$$

  Essentially, this means that with probability one, there will be some time when some agent will produce an optimal solution. This is the weakest of the three convergence

notions: Already ϵ-convergence in solution implies convergence in value, since if (11) is valid for some $\epsilon > 0$, the agents perform independent trials to hit an optimal solution with a success probability that is bounded from below by a positive constant after iteration $M(\epsilon)$, which implies success in *some* iteration with probability one.

Now we are able to cite available results, which can be demonstrated for arbitrary problems on arbitrary construction graphs. In Gutjahr [17], it has been shown that an ACO algorithm called GBAS has the property of ϵ-convergence in solution. The threshold $\epsilon > 0$ can be chosen arbitrarily, but the parametrization of the algorithm has to be adapted to ϵ. GBAS can be described as $\mathcal{MM}\text{AS}_{bs}$ with $\tau_{min} = 0$, $\tau_{max} = \infty$ and pheromone update only in iterations where $\hat{w}$ improves.

Stützle and Dorigo show in [30] that $\mathcal{MM}\text{AS}_{bs}$ and $\mathcal{MM}\text{AS}_{it}$ possess the property of convergence in value. This is a very general result. From their derivations, it also follows that the investigated $\mathcal{MM}$AS algorithms (with some constant $\tau_{min} > 0$) have the property of ϵ-convergence in solution, although ϵ cannot be chosen arbitrarily here, but depends on the setting of τ_{min} and is usually large. Convergence in solution is not possible in this context.

The strongest convergence property, convergence in solution, has been demonstrated in Gutjahr [18] for two variants of $\mathcal{MM}\text{AS}_{bs,co}$, called GBAS/tdev and GBAS/tdlb. The first variant, GBAS/tdev, differs from $\mathcal{MM}\text{AS}_{bs,co}$ by using time-dependent evaporation rates, that is, ρ is replaced by ρ_m, where the values ρ_m are decreasing in m according to a suitable scheme. The second variant, GBAS/tdlb, differs from $\mathcal{MM}\text{AS}_{bs,co}$ by using time-dependent lower bounds, that is, τ_{min} is replaced by $\tau_{min}(m)$, where the values $\tau_{min}(m)$ are decreasing in m, again according to a suitable scheme.[3]

Sebastiani and Torrisi [29] prove an important extension of the results in [18] by applying time-dependent evaporation rates and time-dependent lower bounds to $\mathcal{MM}\text{AS}_{bs}$ instead of $\mathcal{MM}\text{AS}_{bs,co}$.

4 Mathematical Techniques for the Runtime Analysis

In this section, we outline two general techniques that can be helpful for obtaining runtime results on ACO variants. The first technique, *level-reaching estimations*, aims at a direct investigation of the process by which an ACO variant gradually approaches the optimal solution. The method is borrowed from the literature on the analysis of evolutionary algorithms. Despite its simplicity, it turns out to be successful in a surprisingly large number of special situations. Its range of applicability, however, is restricted in the ACO field to variants using exclusively the best-so-far type of reinforcement (e.g., $\mathcal{MM}\text{AS}_{bs}$ or $\mathcal{MM}\text{AS}_{bs,co}$). For being able to cope with ACO variants with other reinforcement mechanisms, we resort to a second group of techniques that *approximate* the pheromone evolution process of a considered ACO variant by mathematically more transparent processes. Of course, such approximations have to be justified by asymptotic convergence results in order to allow conclusions on the behavior of the ACO variant under consideration itself.

For the sake of easier presentation, we will identify "paths" and "solutions" in the sequel and write sometimes also x instead of w.

[3] Contrary to GBAS, pheromone update is done in GBAS/tdev and GBAS/tdlb also in iterations where $\hat{w}$ does not improve, as usual in the $\mathcal{MM}$AS class of algorithms.

4.1 Level-Reaching Estimations

Level-reaching estimations, which we shall describe here in a generic context, have been developed and successfully used in diverse publications on the analysis of evolutionary algorithms. For an excellent example, we refer to Droste et al. [14]. The following introduction of the basic idea builds on the discussion in Borisovsky and Eremeev [5]. There, the authors argue that a straightforward way to the runtime analysis of evolutionary algorithms would be the application of Markov chain theory. However, the explosion of the solution space in the case of growing instance size makes this approach usually infeasible. Therefore, it is necessary to *group* solutions into classes, a suitable criterion for the classification being the fitness of the solutions: Let $f_1, \ldots, f_d$ denote the possible different values the fitness function f can assume on the finite set S, where we suppose that these values are sorted such that $f_1 < \ldots < f_d$. We call $A_j = \{x \in S \mid f(x) = f_j\}$ the jth *level set*. On certain conditions, the search behavior of an evolutionary metaheuristic is in some aspects invariant with respect to the choice of specific elements from a level set A_j. In particular, there may be bounds for the expected staying time in some level set that do not depend on the specific solution(s) visited (first) in this level set, but only on the level number j. This can then be exploited for runtime analysis purposes.

In our ACO framework, we can apply this idea as follows. Consider an ACO variant with BS pheromone update, such as, for example, $\mathcal{MM}\text{AS}_{bs}$ or $\mathcal{MM}\text{AS}_{bs,co}$. Similarly as before, we denote the best-so-far solution (path) by $\hat{x}$. Assume $\hat{x} \in A_j$. By definition, $\hat{x}$ does not change as long as the agents do not produce a better solution than $\hat{x}$. During all this time, only the edges on $\hat{x}$ are reinforced. Now suppose that a lower bound λ_j for the probability can be found that on these premises, one of the agents finds a solution better than $\hat{x}$, that is, a solution with some fitness value f_k where $k > j$. As soon as this happens, the process jumps from level j up to level k. Provided that λ_j only depends on the current level number j, the expected runtime until the process jumps from level j to a higher level has upper bound $1/\lambda_j$. In other words, the expected staying time in level j is then bounded from above by $1/\lambda_j$.

By definition of $\hat{x}$, after $\hat{x}$ has left a level set A_j, it can never return to A_j again, so each level set is visited at most once. In the worst case, $\hat{x}$ visits *all* suboptimal level sets A_j $(j = 1, \ldots, d-1)$ before reaching the optimal level set A_d. Thus, $1/\lambda_1 + \ldots + 1/\lambda_{d-1}$ is an upper bound for the overall runtime of the algorithm until reaching the optimum.[4] In its application to special cases, this general approach may still need some extensions, which we shall not discuss here, since they do not contribute to the main idea.

An obvious condition for the applicability of the outlined level-reaching estimation technique is that the process never returns from a current level set A_j to a *lower* level set A_ℓ with $\ell < k$. Contrary to the best-so-far solution $\hat{x}$, the fitness of the iteration-best solution $\tilde{x}$ can also decrease from one iteration to the next. Therefore, the analysis of variants using the IB instead of the BS reinforcement rule is not amenable to the described technique.

4.2 Approximation Techniques

To overcome the just-mentioned difficulties in the analysis of ACO variants that do not possess anymore the monotonicity properties present in a BS reinforcement context, a quite different analytical approach will be outlined in this subsection. Its origins lie in the stimulating work

[4] Although this observation seems intuitively evident and has indeed been used in several publications, a rigorous formal proof is surprisingly intricate. Such a proof is given in [22].

by Merkle and Middendorf [25] on the *ACO Model*.

4.2.1 ACO Model and Associated Discrete Deterministic Process

The ACO model was introduced in [25] within the context of a specific permutation problem (the linear assigment problem), applying a special ACO variant using IB reinforcement. More recently [20], a similar approach has been used for the analysis of *Ant System* under the name *Associated Discrete Deterministic Process* (ADDP). Making use of the presentation of AS in section 2, it is easier to start here with the explanation of the ADDP. We describe the concept on a quite general level, using again the construction graph framework of section 2. The key idea of the ADDP, borrowed from the ACO model, is the following: ACO is a stochastic procedure, since in each iteration, agents perform random walks on the construction graph. As a consequence, also the pheromone vectors occurring in iteration $1, 2$, etc., are random variables. However, if we consider the case of a growing number S of agents, the effect of their collective behavior on the pheromone increment becomes less and less stochastic because of statistical laws. For AS, let us assume for simplicity that the constant C in (4) is chosen as 1. By the Law of Large Numbers (considering that the random paths of the agents are selected independently of each other), in the limit $S \to \infty$, the actual pheromone increment $(\rho/S) \cdot \sum_{s=1}^{S} \Delta\tau_{ij}^{s}$ in (3) becomes

$$\rho \cdot \frac{1}{S} \sum_{s=1}^{S} \mathbf{I}\left\{ (i,j) \in w^s \right\} \cdot f(w^s) \sim \rho \cdot \mathbf{E}\left[\mathbf{I}\left\{(i,j) \in w^s)\right\} \cdot f(w^s)\right], \tag{13}$$

where $\mathbf{I}$ is the indicator function defined by $\mathbf{I}(A) = 1$ if statement A holds and $\mathbf{I}(A) = 0$ otherwise, and $\mathbf{E}$ denotes the mathematical expectation. The expression

$$F_{ij}(\tau) = \mathbf{E}\left[\mathbf{I}\left\{(i,j) \in w(\tau)\right\} \cdot f(w(\tau))\right] \tag{14}$$

where $w(\tau)$ is (as in section 3) the random walk of an agent based on current pheromone vector τ, has been called *expected passage fitness* of edge (i,j) in [20]. Eq. (13) suggests to approximate the random pheromone increment for edge (i,j) in AS by the deterministic quantity $\rho \cdot F_{ij}(\tau)$. If this is done in each iteration, we obtain the process called ADDP in [20], formally defined by the pheromone update rule

$$\tau_{ij} := (1-\rho) \cdot \tau_{ij} + \rho \cdot F_{ij}(\tau). \tag{15}$$

The article [25] on the *ACO Model* by Merkle and Middendorf was the first to introduce the idea of approximating the pheromone increment by its expected value. The ACO model concept as developed in [25] is very similar to that of the ADDP described above, with the exception that the pheromone update in the ACO model uses an iteration-best rule instead of the update rule of AS, which leads to slightly different formulas.

The approximation performed in the ADDP can be justified by an asymptotic convergence result: Theorem 2.1 in [20] shows that during each period containing a fixed number of iterations, with a probability of at least $1 - \epsilon$ for some arbitrary $\epsilon > 0$, the pheromone vectors of the ADDP deviate from those of AS in each iteration only by less than some arbitrary $\delta > 0$ (where deviation is measured by the Euclidean distance), provided that (a) the number S of agents is chosen sufficiently high in dependence of ϵ, δ and the period length, and (b) some technical conditions concerning the fitness function are satisfied.

4.2.2 Associated Continuous Deterministic Process and Theory of Slow Learning

The concepts of the ACO model and of the ADDP already facilitate the investigation of certain aspects of ACO algorithms. However, the picture they provide may still be too complex to enable a mathematical runtime analysis. To give a clearer view on the fundamental dynamics of the considered processes, the asymptotic case of an evaporation rate ρ near zero can be studied. This asymptotic case is motivated by the fact that in the results reported in section 3, decreasing ρ values were shown to be advantageous for convergence to the optimal solution. It is obvious that if ρ is reduced, such that pheromone changes in each iteration become small, a comparably larger number of iterations must be executed to get substantial changes. Therefore, in order to be able to plot the asymptotic behavior, it seems convenient to re-scale the time axis in such a way that the product of ρ and the number M of iterations per time unit remains constant. Without loss of generality, we can assume that the product is one, that is, we assume that $\rho = 1/M$. In this scaling, an iteration takes $dt = 1/M = \rho$ time units.

Based on this consideration, the *Associated Continuous Deterministic Process* (ACDP) has been defined in [20] as the limit of the (re-scaled) ADDP as $dt \to 0$: With $\tilde{\tau}(t) = (\tilde{\tau}_{ij}(t))$ denoting the pheromone vector at time t, eq. (15) can now be written as

$$\tilde{\tau}_{ij}(t + dt) = \tilde{\tau}_{ij}(t) + dt \cdot \{-\tilde{\tau}_{ij}(t) + F_{ij}(\tilde{\tau}(t))\}.$$

Letting $dt \to 0$, we obtain that the ACDP is given by the system

$$\frac{d\tilde{\tau}_{ij}}{dt} = F_{ij}(\tilde{\tau}) - \tilde{\tau}_{ij} \qquad ((i,j) \in \mathcal{A}) \tag{16}$$

of ordinary differential equations, where $F_{ij}(\tilde{\tau})$ is the expected passage fitness of edge (i,j) under $\tilde{\tau}$. The difference on the right hand side plays already an important role in [25] as it describes the drift toward the fixed point in the analysis of the ACO model when using one ant only.

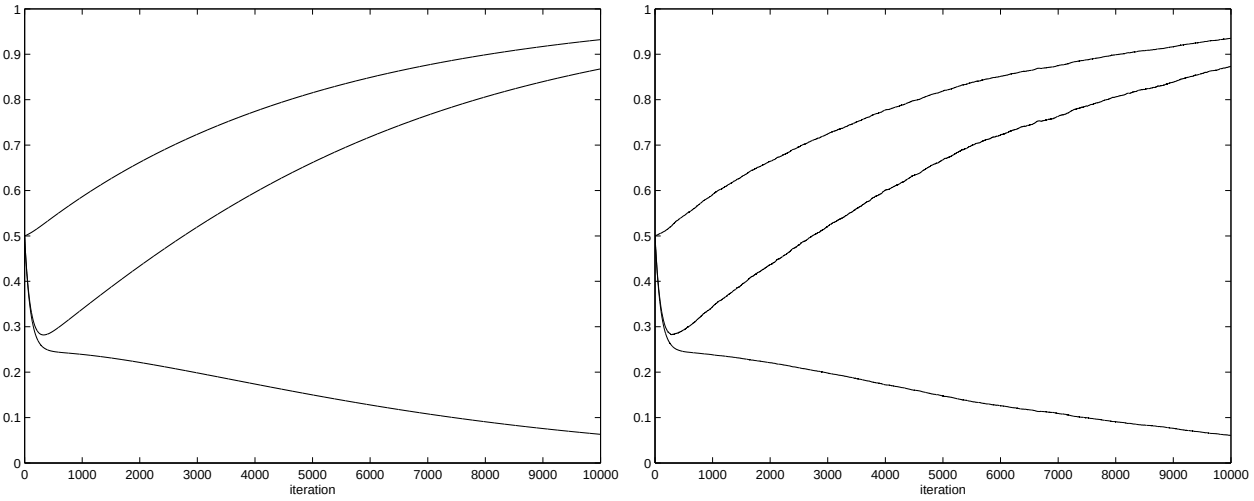

Fig. 4. Theoretical prediction (left plot) and real run (right plot) of AS on a OneMax problem with $n = 50$, $S = 50$ and $\rho = 0.01$. From top to bottom, the curves represent average relative fitness, pheromone trail for up-moves and pheromone trail for down-moves.

Also for the ACDP, the question of whether the approximation can be justified by a strict asymptotic convergence result is of interest. It turns out that the theorem cited at the end of subsection 4.2.1 cannot immediately be extended to the ACDP case. The reason is that after re-scaling the time axis, a given period containing a fixed number of iterations (i.e., the period for which the mentioned theorem ensures uniform asymptotic convergence) *shrinks* on the re-scaled axis with decreasing ρ values. Thus, in the ACDP framework, the theorem only provides a *local* convergence behavior, which is not sufficient for obtaining statements on the overall runtime until hitting an optimal solution.

Fortunately, it is nevertheless possible to derive asymptotic convergence results ensuring "closeness' of the trajectories of the ACDP to those of AS. The mathematical key to such proofs is provided by a rather old theorem by Normans [28], originally developed for investigations in mathematical psychology. Norman investigated the properties of systems where *learning* takes place. In particular, he considered the asymptotic case (called the case of *slow learning*) where the learning rate is comparably low, but the number of learning events is high. Now, the learning rate in this theory corresponds exactly to the evaporation rate ρ in ACO, such that the approximation assumptions of the ACDP (low ρ and large number of iterations) are identical to those in the theory of slow learning.

Because of space limitations, we do not reproduce here Norman's [28] fundamental theorem on slow learning, but refer the reader to [21] for a presentation within an ACO context and an exemplary application to the runtime analysis of AS on OneMax. Fig. 4 illustrates the good coincidence between predictions based on the ACDP approximation and the real behavior of AS.

5 Some Available Runtime Results

Almost all available runtime results in the field of evolutionary algorithms (EA) refer to the optimization of *pseudo-boolean* fitness functions. A pseudo-boolean function is a function f mapping the set $S = \{0,1\}^n$ of binary vectors of length n into the set of real numbers. As shown in Example 2 in section 2, subset selection problems have a natural interpretation as problems with pseudo-boolean fitness functions. In order to be able to compare the performance of ACO algorithms to that of diverse evolutionary algorithms, the runtime analysis of ACO variants should start with an investigation of fitness functions of this type. This aim has also been formulated as the "Open Problem 1" of ACO theory in Dorigo and Blum [8]. Only very simple test functions have been fully analyzed up to now. In particular, let us consider the following problems, taken from the EA literature (e.g., [24], [14]):

- **Needle in a Haystack (NH) problem:**

$$f(x) = \mathbf{I}\{x = x^*\} \to \max, \quad x \in \{0,\, 1\}^n, \tag{17}$$

  where $x^* \in \{0,\, 1\}^n$ is a fixed solution, and $\mathbf{I}$ denotes again the indicator function.

- **k-Needles in a Haystack (k-NH) problem:**

$$f(x) = \mathbf{I}\{x = x_1^* \text{ or } \ldots \text{ or } x = x_k^*\} \to \max, \quad x \in \{0,\, 1\}^n, \tag{18}$$

  where $x_1^*, \ldots, x_k^* \in \{0,\, 1\}^n$ are k different fixed solutions.

- **Generalized OneMax problem.**

$$f(x) = n + c - d(x, x^*) \to \max, \quad x \in \{0,\, 1\}^n, \tag{19}$$

where $x^* \in \{0,\, 1\}^n$ is a fixed solution, d denotes the Hamming distance and c is a nonnegative constant. For $x^* = (1, \ldots, 1)$ and $c = 0$, this gives the (ordinary) OneMax function $f(x) = \sum_{i=1}^{n} x_i$.

In some sense, NH and OneMax represent in an idealized way opposite possible characteristics of a more complex fitness function: OneMax stands for the case where a fitness function gives good "guidance" to the search process, whereas NH represents the case where such guidance is completely lacking. In order to have also an example in between, we will additionally consider a combination of NH and OneMax defined as follows:

- **NH-OneMax problem**: Let integers n and $k \leq n$ be given.

$$f(x) \;=\; \left(\prod_{i=1}^{k} x_i\right) \cdot \left(\sum_{i=k+1}^{n} x_i + 1\right) \;\to\; \max, \quad x \in \{0,1\}^n. \tag{20}$$

This function consists of a NH part (connected with the first k bits of the solution x) for which the correct solution has to be found without guidance, that is, by trial-and-error. Only provided that the NH part is solved, the fitness becomes positive. The second part (connected with the remaining $n - k$ bits) consists in the solution of a OneMax problem.

5.1 Ant System

For AS, two types of investigations have been started. The first is a comparison of the performance of AS on different possible construction graphs, with the aim of obtaining information on the most suitable choice by analytical means. The second is an investigation of the runtime complexity, that is, a study of the influence of the problem size n on the (order of the) runtime needed for finding the optimal solution.

5.1.1 Comparison of Construction Graphs

In [20], the runtime performance of AS on the *chain* construction graph, the *drum* construction graph (both explained in section 2) and a third construction graph called *disk* is compared by means of the ACDP approach outlined in subsection 4.2.2. The disk (which lets the agents randomly gather 1-bits and leave the construction graph at some time) proves as not competitive—and even seriously inefficient—in the case of the NH problem. Intuitively speaking, the reason is that whereas chain and drum have "built-in" stopping criteria, agents on the disk have to "learn when to stop", which is susceptible to pre-mature convergence. For generalized OneMax, chain and drum are equally good in their runtime behavior. An advantage of the drum over the chain can be verified analytically as soon as fitness functions with more than one local optimum are considered, which is the case for 2-NH.

5.1.2 Runtime Complexity on Generalized OneMax

In [19], [21], the runtime of AS in dependence of the problem size n using the *chain* construction graph has been investigated for the generalized OneMax problem (19). For the analysis, the ACDP approach has been applied. In the considered context, it turns out that the asymptotic equivalence of AS and ACDP can be justified by a strict mathematical proof. For the corresponding ACDP, it can be shown that for each $\epsilon > 0$, the required amount of (re-scaled) time t until the expected relative fitness[5] of the current solution x reaches a value of $1 - \epsilon$, is bounded from above by $2(n + c)\log((1 - \epsilon)/\epsilon)$ and from below by $(n/2)\log((1 - \epsilon)/\epsilon)$. By choosing $\epsilon = \epsilon_0/n$ for a small constant ϵ_0, we see that the expected runtime of the ACDP until achieving an *absolute* deviation from the maximum fitness value n of OneMax by less than ϵ_0 is of order $\Theta(n \log n)$, where Θ refers to an upper *and* lower bound. This order is the same as in the well-known results on the runtime complexity of the (1+1)-Evolutionary Algorithm (short: (1+1)-EA) on OneMax (see [14]).

Two remarks on this result are in place: First, it must be emphasized that the result holds for an approximation to AS instead of AS itself. Despite the mentioned mathematical convergence result, there remains a difference between the two processes unless we would let $\rho = \rho(n)$ tend to 0 with increasing instance size n, which, however, would increase the runtime complexity because ρ is also used as a scaling factor between AS and ACDP. In this respect, the situation is similar to that in classical statistical test theory, where tests are derived from asymptotic approximations to test statistics distributions: Making the applied formulas *exact* would require to let the sample size and hence the experimental effort go to infinity. This cannot be done in practice, so the mentioned statistical results are, in a strict mathematical sense, not "valid" for the situations to which they are applied. Nevertheless, their usefulness is not doubted.

Secondly, it should be noted that compared to the (1+1)-EA results in the literature, the result on AS in [21] refers to the stronger notion of convergence in solution (see eq. (10)): Given that AS is suitably approximated by the ACDP, the agents are not only able to hit the solution (by chance) for the first time within $O(n \log n)$ time, but even to *learn* the solution by storing its characteristics in the pheromone vector within this time.

5.2 $\mathcal{MM}$AS

Let us now consider $\mathcal{MM}$AS variants instead of AS, in particular $\mathcal{MM}\text{AS}_{bs,co}$ and $\mathcal{MM}\text{AS}_{bs}$.

5.2.1 Generalized OneMax

The work [19], [21] contains also an analysis of the runtime complexity of $\mathcal{MM}\text{AS}_{bs,co}$ working on the *chain* construction graph for Generalized OneMax. The main result is the following: Again expressed by the expected number of function evaluations until reaching the optimal solution, the expected runtime of $\mathcal{MM}\text{AS}_{bs,co}$ with $\rho = 1 - a/n$, $\tau_{min} = a/n^2$ ($a > 0$ constant), $C = 1/n$ and $\tau_0 = 1/(2n)$, applied to (19), is bounded from above by $(2e^{2a}/a) \cdot nH_n$ for $n \geq 2a$, where $H_n = \sum_{j=1}^{n} j^{-1}$ is the nth harmonic number. This bound is of order $O(n \log n)$. The proof of the result consists in a rather straightforward application of the level-reaching estimation technique described in subsection 4.1.

[5] The relative fitness of x is defined as $(f(x) - f_{min})/(f_{max} - f_{min})$, where $f_{min} = \min_{x \in S} f(x)$ and $f_{max} = \max_{x \in S} f(x)$.

5.2.2 Low and High Evaporation Rates

There is a special interesting point in the result cited in subsection 5.2.1: Contrary to what convergence results (section 3) may indicate as favorable, and contrary to the asymptotic situation considered in the analysis of AS (subsection 5.1), the result cited above holds for *high* evaporation rate values ρ, even values that tend to the maximum value 1 as the problem size n increases. This can make us suspect that for $\mathcal{MM}$AS variants of ACO, the evaporation rate should be chosen not close to zero, but rather close to one.

A similar conclusion is suggested by the results provided by Neumann and Witt in [26]. The authors consider a variant derived from GBAS [17], [18] they call 1-ANT. The used construction graph is the *chain*. Compared with [18], not only lower pheromone bounds, but also upper pheromone bounds are used, and pheromone is re-normalized explicitly after each update to a sum of 1 over all edges of the construction graph, the re-normalized values being stored as the new pheromone values.[6] To be more specific, the authors use the pheromone update formula

$$\bar{\tau}_{ij} := \left[\frac{(1-\bar{\rho}) \cdot \bar{\tau}_{ij} + \bar{\rho} \cdot \mathbf{I}\{(i,j) \in \hat{w}\}}{1 - \bar{\rho} + 2n\bar{\rho}} \right]_{\bar{\tau}_{min}}^{\bar{\tau}_{max}}, \tag{21}$$

where $\bar{\tau}_{min} = 1/(2n^2)$ and $\bar{\tau}_{max} = (n-1)/(2n^2)$. We write $\bar{\rho}$ and $\bar{\tau}$ here instead of ρ and τ, respectively, in order to be able to show the connections to $\mathcal{MM}AS$ below. In 1-ANT, the best-so-far solution $\hat{w}$ is updated not only if a better new solution is found, but it is replaced also by an equally good new solution, contrary to the usual procedure in $\mathcal{MM}AS$ (see, e.g., [30], cf. also [18]). Only one single agent is used (i.e., $S = 1$).

The authors show that there is a large range of evaporation rate values for which the behavior of 1-ANT is identical to that of (1+1)-EA on *all* problems with pseudo-boolean fitness functions. More precisely, this equivalence holds for all values $\bar{\rho} \geq (n-2)/(3n-2)$.

For values $\bar{\rho}$ tending to zero with increasing n (for which the mentioned equivalence does not hold), Neumann and Witt investigate the runtime explicitly for the case of OneMax. In evaporation rate schemes for which the behavior of 1-ANT is different from (1+1)-EA, competitiveness to (1+1)-EA could *not* be confirmed: For $\bar{\rho} = \Omega(n^{-1+\epsilon})$ with some constant $\epsilon > 0$, it was shown that the optimization time of 1-ANT on OneMax is $O(n^2)$ with a probability $1 - 2^{-\Omega(n^{\epsilon/2})}$. Evidently, the bound is worse than the expected runtime of (1+1)-EA. Moreover, as long as this result is not combined with a suitable bound on the runtime in the exceptional cases with probability $2^{-\Omega(n^{\epsilon/2})}$, it admits no conclusion on the *expected* runtime.[7]

For $\bar{\rho} = O(n^{-1-\epsilon})$ with some constant $\epsilon > 0$, the runtime of 1-ANT on OneMax is $2^{\Omega(n^{\epsilon/3})}$ with a probability $1 - 2^{-\Omega(n^{\epsilon/3})}$. This implies that the runtime for such a scheme of $\bar{\rho}$ values grows exponentially in the instance size.

In total, competitiveness of 1-ANT with (1+1)-EA on OneMax is only obtained in cases where they collapse.

5.2.3 1-ANT and Standard $\mathcal{MM}$AS

An interesting question is whether the behavior described above also holds for $\mathcal{MM}$AS where (as in most ACO variants) no explicit re-normalization of pheromone is done[8], and where the

[6] This is in a similar vein as the re-normalization done in the ACO Hyper-Cube Framework [3].

[7] The symbol $\Omega(h(k))$ refers to a lower bound of the form $c \cdot h(k)$.

[8] Recently, Birattari et al. [2] showed that the most frequently applied ACO variants, including AS and $\mathcal{MM}$AS, are in a certain sense *scale-invariant*, which implies that pheromone trail re-normalization is not

best-so-far solution is only updated if a *strictly* better new solution is found. Furthermore, also fitness-proportional reward might be investigated in addition to constant reward. In Gutjahr and Sebastiani [22], the expected runtime of the two algorithms $\mathcal{MM}\text{AS}_{bs,co}$ and $\mathcal{MM}\text{AS}_{bs}$ is analyzed for the test problems NH, (Generalized) OneMax, NH-OneMax and LeadingOnes. In contrast to the investigation in [19] and [21], the $\mathcal{MM}$AS variants are now also given an upper pheromone bound. The used construction graph is the *chain*. Lower and upper pheromone bounds are allowed to depend on instance size n, and the two bounds are chosen in a symmetric fashion: $\tau_{max} = 1 - \tau_{min}$. Similarly as τ_{min}, also the evaporation rate ρ is allowed to depend on n, which spans a rich class of possible schemes combining the sequences $(\tau_{min}(n))$ and $(\rho(n))$.

Let us outline some of the obtained results. For small τ_{min}, the considered $\mathcal{MM}$AS variants degenerate to (1+1)-EA only in a marginal domain of evaporation rate values close to one:[9] This degeneration takes place if $\rho \geq 1 - \tau_{min}/\tau_{max}$. In the Generalized OneMax case, an $O(n \log n)$ expected runtime complexity is obtained for a large variety of $(\tau_{min}(n), \rho(n))$ schemes. For example, for $\mathcal{MM}\text{AS}_{bs,co}$, the scheme $\tau_{min} = 1/n$, $\rho = const$ leads to $O(n \log n)$ expected runtime, also if ρ is chosen very small. This observation is important in so far as it demonstrates that we are not dependent on high ρ values to achieve a behavior of $\mathcal{MM}$AS on OneMax that is competitive with (1+1)-EA. Even the case $\tau_{min} = 1/n$ and $\rho = 1/n$ still leads to an expected runtime of $O(n^2)$. Similar results are obtained for $\mathcal{MM}\text{AS}_{bs}$.

Moreover, it turns out that working with small evaporation rates for $\mathcal{MM}$AS has a considerable *advantage* compared to large values. This becomes evident when test functions are investigated that do not give suitable guidance for the search anymore (as given by OneMax). The simplest example is NH. Here, both (1+1)-EA and $\mathcal{MM}$AS parametrizations with high ρ perform very poorly; decreasing ρ can considerably improve the runtime.[10]

It is especially instructive to look at the combination of "guiding" and "non-guiding" features in the test function NH-OneMax (see eq. (20)). For $k = \log_2 n$, both (1+1)-EA and $\mathcal{MM}$AS with high ρ have in this case *exponential* expected runtime, whereas a scheme with $\rho(n) = n^{-3}$ shows *polynomial* expected runtime. This seems to give a definite hint that also in the $\mathcal{MM}\text{AS}_{bs}$ framework, comparably low values for the evaporation rate ρ should be used, and that the size of ρ should possibly even be decreased with increasing instance size.[11]

For the LeadingOnes test function (cf. [14]), the investigation in [22] shows an expected runtime of order $O(n^2)$ as long as ρ is of order $\Omega((\log n)/n)$.

Let us give a closer look at the differences between the algorithms 1-ANT and $\mathcal{MM}$AS analyzed in [26] and [22], respectively. First of all, notation should be made comparable. By

necessary, at least not for scaling reasons.

[9] For the sake of comparability with $\mathcal{MM}\text{AS}_{bs}$, where a new solution is only accepted if it is better than the currently best solution (and not just equally good), [22] considers an (1+1)-EA variant following the same rule. Several publications on (1+1)-EA deal with a variant where also equally good solutions are accepted.

[10] Intuitively, in the case of very small ρ, pheromone remains nearly constant (identical to the initial values), which leads to a pure random search behavior. In the case of high ρ, on the other hand, the current solution has a considerable influence (and, as discussed in [22], for some problems a too large influence) on the next solution to be constructed, since it determines whether the pheromone trail of a bit is at the upper or at the lower limit. This leads to a kind of "aggressive search", which is favorable for fitness functions providing guidance and unfavorable for not-guiding or even deceptive fitness functions.

[11] It is important to keep in mind that this consideration only refers to best-so-far reinforcement schemes. Empirical evidence seems to indicate that the recommendation to decrease ρ with increasing n is not valid for the case of iteration-best reinforcement.

the transformation

$$\rho = \frac{2n\bar{\rho}}{1 - \bar{\rho} + 2n\bar{\rho}}, \quad \tau_{ij} = 2n\bar{\tau}_{ij},$$

eq. (21) can be re-written as

$$\tau_{ij} := \left[(1-\rho)\,\tau_{ij} + \rho \cdot \mathbf{I}\{(i,j) \in \hat{w}\}\right]_{\tau_{min}}^{\tau_{max}},$$

with $\tau_{min} = 1/n$ and $\tau_{max} = 1 - 1/n$, which is the usual $\mathcal{MM}\text{AS}_{bs,co}$ dynamics investigated also in [22]. Nevertheless, also using this transformation, the results in [26] and [22] are not equivalent except for $\rho \geq \tau_{min}/\tau_{max} = 1 - 1/(n-1)$, which corresponds to $\bar{\rho} \geq (n-2)/(3n-2)$, where both algorithms coincide with (1+1)-EA. For example,

- for constant ρ, where $\mathcal{MM}\text{AS}_{bs,co}$ has expected runtime $O(n \log n)$, the corresponding value for $\bar{\rho}$ is of the "threshold" order $\Theta(n^{-1})$ for which no statement on the behavior of 1-ANT is made in [26], and
- for $\rho = n^{-1}$, where $\mathcal{MM}\text{AS}_{bs,co}$ has expected runtime $O(n^2)$, the corresponding value $\bar{\rho}$ is of order $\Theta(n^{-2})$, that is, already in the area where 1-ANT has exponential expected runtime.

The reason seems to lie in the different update rules for the best-so-far solution: exchanging $\hat{x}$ also in cases of an equally good new solution, as it is done in 1-ANT, obviously deteriorates the performance of the algorithm on OneMax.

For LeadingOnes, the situation appears to be quite similar as for OneMax: In a recent technical report by Doerr et al. [7], an upper bound of order $O(n^2 \cdot 2^{9/(n\bar{\rho})})$ for the expected runtime of 1-ANT on LeadingOnes is derived; this bound is polynomial for $\bar{\rho} = \Omega(1/(n \log n))$ and only $O(n^2)$ for $\bar{\rho} = \Omega(1/n)$, and it is superpolynomially large for $\bar{\rho} = o(1/(n \log n))$. The scheme $\rho = \Omega((\log n)/n)$, for which $\mathcal{MM}\text{AS}_{bs,co}$ has expected runtime $O(n^2)$ on LeadingOnes according to [22], corresponds to $\bar{\rho} = \Omega((\log n)/n^2)$ with superpolynomial expected runtime for 1-ANT. Possibly, test functions giving few guidance might lead to the opposite effect; thus, it would be interesting to have results for 1-ANT on NH and NH-OneMax.

In addition to the LeadingOnes results, [7] also contains a result on the BinVal function: The authors derive a bound of order $O(n^2 \cdot 2^{O((\log^2 n)/(n\bar{\rho}))})$ for the expected runtime of 1-ANT for this test function.

5.2.4 Further results

Finally, a recent technical report by Neumann and Witt [27] might be mentioned where an ACO variant is investigated which the authors call again 1-ANT, although both solution construction and pheromone update are handled in a quite different way, compared to [26]. Two types of construction graphs are used, based on two classical problem-specific algorithms for the MST: the algorithm by Broder and that by Kruskal, respectively. In both cases, pheromone is restricted to only two values ℓ and h. Between these two values, the algorithm switches governed by problem-specific criteria which are not derived here from fitness, but rather from graph structure and constraints. A version where only heuristic information is used ($\alpha = 0$ and β large enough) is shown to be able to mimic Kruskal's algorithm.

6 Conclusions

In this article, we have presented a unified framework for the theoretical runtime analysis of ACO algorithms, described some formal techniques that may be helpful for investigations of this type, and outlined first available results. In particular, we have recalled results for ACO variants allowing a comparison with the evolutionary algorithm that has found most interest in theoretical research, (1+1)-EA. Moreover, the results available now already give some hints on the appropriate choice of an important parameter, the evaporation rate.

Nevertheless, the present state of the mathematical runtime analysis of ACO algorithms is still characterized by the fact that there exist many more open problems than solved questions. To give a few examples: Contrary to $\mathcal{MM}\text{AS}_{bs}$, there seem to be no analytical results available at the moment on the runtime behavior of $\mathcal{MM}\text{AS}_{ib}$ and ACO_{rank}. Possibly, the investigation of these variants could start with $\mathcal{MM}\text{AS}_{ib}$ without bounds, which coincides with ACO_{rank} with $R = 2$. Furthermore, applied to pseudo-boolean functions and using the chain, both variants also coincide with PBIL (*Popoulation-Based Incremental Learning*, see [1]). Some first analytical investigations relating to runtime results in the PBIL field have already appeared (see, e.g., [15], [16]). Maybe there can be a cross-fertilization between the work in the PBIL area and that in ACO.[12] Also for ACO variants as ACS or HC-ACO, no analytical runtime results are known up to now.

Of course, investigations should be extended to more complex test problems, including so-called *deceptive* problems (see, e.g., [4]). An important further issue for future research is the transfer of the mentioned analytical results on pseudo-boolean functions to routing, sequencing or scheduling problems where "traditional" construction graphs (as that in Fig. 1) are applied. Although there is no doubt that such investigations will be rather challenging, an analysis of these problems should be manageable.

References

[1] Baluja, S., Caruana, R., "Removing the genetics from the standard genetic algorithm", *Proc. of the 12th Int. Conf. on Machine Learning* (ML-95), eds.: A. Prieditis and S. Russell, Morgan Kaufmann Publishers, Palo Alto, CA, pp. 38–46 (1995).

[2] Birattari, M., Pellegrini, P., Dorigo, M., "On the invariance of ant colony optimization", *IEEE Transactions on Evolutionary Computation* (in press) (2007).

[3] Blum, C., Dorigo, M., "The hyper-cube framework for ant colony optimization", *IEEE Trans. on Systems, Man, and Cybernetics – Part B*, 34 (2), pp. 1161–1172 (2004).

[4] Blum, C., Dorigo, M., "Search bias in ant colony optimization: on the role of competition-balanced systems", *IEEE Transactions on Evolutionary Computation* 9, no. 2, pp. 159–174 (2005).

[5] Borisovsky, P.A., Eremeev, A.V., "A study on performance of the (1+1)-Evolutionary Algorithm", *Proc. Foundations of Genetic Algorithms* 7, Morgan Kaufmann: San Francisco, CA (2003).

[12] The close relations between ACO and certain Estimation-of-Distribution (EDA) algorithms such as PBIL have also been outlined in Zlochin et al. [34] by the unifying concept of *model-based search*.

[6] Bullnheimer, B., Hartl, R. F., Strauss, C., "A new rank–based version of the Ant System: A computational study", *Central European Journal for Operations Research and Economics* 7 (1), pp. 25–38 (1999).

[7] Doerr, B., Neumann, F., Sudholdt, D., Witt, C., "On the influence of pheromone updates in ACO algorithms", Technical Report CI-223/07, University of Dortmund, SFB 531 (2007), to appear in Proc. GECCO 2007.

[8] Dorigo, M., Blum, C., "Ant colony optimization theory: a survey", *Theoretical Computer Science* 344, pp. 243–278 (2005).

[9] Dorigo, M., Di Caro, G., "The Ant Colony Optimization metaheuristic", in: *New Ideas in Optimization*, D. Corne, M. Dorigo, F. Glover (eds.), pp. 11-32, McGraw–Hill: Maidenhead, UK (1999).

[10] Dorigo, M., Gambardella, L.M., "Ant Colony System: A cooperative learning approach to the traveling salesman problem", *IEEE Transactions on Evolutionary Computation* 1, pp. 53–56 (1997).

[11] Dorigo, M., Maniezzo, V., Colorni, A., "Positive feedback as a search strategy", Technical Report TR-POLI-91-016, Dipartimento di Elletronica, Politecnico di Milano, Milan, Italy (1991).

[12] Dorigo, M., Maniezzo, V., Colorni, A., "Ant System: Optimization by a colony of cooperating agents", *IEEE Trans. on Systems, Man, and Cybernetics* 26, pp. 1–13 (1996).

[13] Dorigo, M., Stützle, T., *Ant Colony Optimization*, MIT Press, Cambridge, MA (2004).

[14] Droste, S., Jansen, T., Wegener, I., "On the analysis of the (1+1) evolutionary algorithm", *Theoretical Computer Science* 276, pp. 51–81 (2002).

[15] Gonzalez, C., Lozano, J.A., Larrañaga, P., "Analyzing the PBIL algorithm by means of discrete dynamical systems", *Complex Systems* 4, pp. 465–479 (2000).

[16] Gonzalez, C., Lozano, J.A., Larrañaga, P., "The convergence behavior of the PBIL algorithm: a preliminary approach", *Proc. 5th Int. Conf. on Artificial Neural Networks and Genetic Algorithms* ICANNGA '01, pp. 228–231, Springer: Berlin, Germany (2001).

[17] Gutjahr, W.J., "A graph–based ant system and its convergence", *Future Generation Computer Systems* 16, pp. 873–888 (2000).

[18] Gutjahr, W.J., "ACO algorithms with guaranteed convergence to the optimal solution", *Information Processing Letters* 82, pp. 145–153 (2002).

[19] Gutjahr, W.J., "Theory of ant colony optimization: status and perspectives", *MIC '05 (6th Metaheuristics International Conference)*, Proceedings CD-ROM (2005).

[20] Gutjahr, W.J., "On the finite-time dynamics of ant colony optimization", *Methodology and Computing in Applied Probability* 8, pp. 105–133 (2006).

[21] Gutjahr, W.J., "First steps to the runtime complexity analysis of ant colony optimization", *Computers and Operations Research*, in press (available online at Elsevier, 22nd Jan. 2007).

[22] Gutjahr, W.J., Sebastiani, G., "Runtime Analysis of Ant Colony Optimization", Technical Report, Consiglio Nazionale delle Ricerche, Rome (2007), available under: http://www.mat.uniroma1.it/people/sebastiani/preprints.htm.

[23] Jones, T., Forrest, S., "Fitness distance correlation as a measure of problem difficulty for genetic algorithms", *Proc. 6th Int. Conf. on Genetic Algorithms*, pp. 184–192, Morgan Kaufmann: San Mateo, CA (1995).

[24] Kallel, L., Naudts, B., Reeves, C.R., "Properites of fitness functions and search landscapes", in: *Theoretical Aspects of Evolutionary Computing* (eds.: Kallel, Naudts, Rogers), Springer: Berlin, Germany, pp. 174–206 (1998).

[25] Merkle, D., Middendorf, M., "Modeling the dynamics of ant colony optimization", *Evolutionary Computation* 10, pp. 235–262 (2002).

[26] Neumann, F., Witt, C., "Runtime analysis of a simple ant colony optimization algorithm", *Proc. ISAAC '06*, LNCS 4288, pp. 618–624, Springer: Berlin, Germany (2006).

[27] Neumann, F., Witt, C., "Ant colony optimization and the minimum spanning tree problem", Technical Report CI-220/06, University of Dortmund, SFB 531 (2006).

[28] Norman, F., *Markov Processes and Learning Models*, Academic Press, New York and London (1972).

[29] Sebastiani, G., Torrisi, G.L., "An extended ant colony algorithm and its convergence analysis", *Methodology and Computing in Applied Probability* 7, pp. 249–263 (2005).

[30] Stützle, T., Dorigo, M. "A short convergence proof for a class of ACO algorithms", *IEEE Transactions on Evolutionary Computation* 6, pp. 358–365 (2002).

[31] Stützle, T., Hoos, H.H., "MAX-MIN Ant System and local search for the travelling salesman problem", in: T. Baeck, Z. Michalewicz and X. Yao (eds.), *Proc. ICEC '97* (Int. Conf. on Evolutionary Computation), pp. 309–314, IEEE Press: New York, NY (1997).

[32] Stützle, T., Hoos, H.H., "MAX-MIN Ant System", *Future Generation Computer Systems* 16, pp. 889–914 (2000).

[33] Wegener, I., Witt. C., "On the analysis of a simple evolutionary algorithm on quadratic pseudo-boolean functions", *J. of Discrete Algorithms* 3, pp. 61–78 (2005).

[34] Zlochin, M., Birattari, M., Meuleau, N., Dorigo, M., "Model-based search for combinatorial optimization: a critical survey", *Annals of Operations Research* 131, pp. 373–379 (2004).