Runtime Analysis of Ant Colony Optimization with Best-So-Far Reinforcement

Walter J. Gutjahr and Giovanni Sebastiani

Abstract: The paper provides some theoretical results on the analysis of the expected time needed by a class of Ant Colony Optimization algorithms to solve combinatorial optimization problems. A part of the study refers to some general results on the expected runtime of the considered class of algorithms. These results are then specialized to the case of pseudo-Boolean functions. In particular, three well known functions and a combination of two of them are considered: the OneMax, the Needle-in-a-Haystack, the LeadingOnes, and the OneMax-Needle-in-a-Haystack. The results obtained for these functions are also compared to those from the well-investigated (1+1)-Evolutionary Algorithm. The results shed light on a suitable parameter choice for the considered class of algorithms. Furthermore, it turns out that for two of the four studied problems, the expected runtime for the considered class, expressed in terms of the problem size, is of the same order as that for (1+1)-Evolutionary Algorithm. For the other two problems, the results are significantly in favour of the considered class of Ant Colony Optimization algorithms.

1 Introduction

Ant Colony Optimization (ACO) has developed in the last fifteen years to one of the most frequently used metaheuristic techniques for solving "hard" combinatorial optimization problems. Numerous publications in a large variety of fields demonstrate the broad applicability and the good performance of this approach. ACO developed from the seminal works by Dorigo, Maniezzo and Colorni [5, 6], which were originally motivated by the attempt to solve the well-known Travelling Salesperson Problem (TSP). Already soon, the inventors of the technique recognized that the approach could be applied to a much larger range of optimization problems. This led to the development of the ACO *metaheuristic*, elaborated by Dorigo and Di Caro in [4]. A recent survey on ACO is given in the excellent book by Dorigo and Stützle [7].

Despite a huge number of 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 Gutjahr [9]-[11] and Stützle and Dorigo [20]). In the meantime, the convergence behavior is already well-explored for a good deal of ACO variants. However, even if an ACO algorithm is ensured to converge to the optimal solution with probability one, there remains the question of how much time it takes (e.g., in the average) until the solution is found. At the moment, results concerning the last question are still scarce in the ACO area.

When proceeding from the question of convergence to that of the *speed* of convergence, one cannot expect anymore to obtain general positive results. This is a consequence of NP-completeness theory and the so-called "no-free-lunch theorems" which state that for each algorithm that performs very well on some optimization problems, there must be other problems

where it fails to be efficient. Runtime analysis of a metaheuristic algorithm has therefore to be done on a detailed level of the investigation of various specific test problems. These test problems may seem simplistic from the viewpoint of applied research, but their investigation can provide helpful information nevertheless in so far as each of them incorporates some typical feature (or several features) of real problems, such that their separate analysis helps us to *understand* why and on which conditions certain algorithmic variants or parametrizations work resp. do not work in real-life applications.

In the present paper, we study the runtime behavior of the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System ($\mathcal{MM}$AS) developed by Stützle and Hoos [21, 22] in the version where it uses best-so-far reinforcement. The runtime is investigated in dependence of the problem size for four specific example problems. Three of them, the OneMax problem, the LeadingOnes problem and the Needle-in-a-Haystack (NH) problem, are well-known and well-analyzed test problems in the Genetic Algorithms (GA) literature. OneMax is a typical example for a problem where the fitness function gives helpful (and never misleading) information supporting the search for the optimal solution. Contrary to that, in the Needle-in-a-Haystack problem, no information of this type is provided. In order to have also a look at a mixture between these two extreme situations, we analyze, as a fourth problem, a combination NH-OneMax of Needle-in-a-Haystack with OneMax.

The $\mathcal{MM}$AS algorithm studied here is basically identical to the variant called GBAS/lb investigated in [14], with the exception that upper bounds for the so called *pheromone values* are used in addition to lower bounds. For GBAS/lb and the related algorithms GBAS, GBAS/tdlb and GBAS/tdev, strong theoretical convergence properties can been shown (see [9]-[11]). Similar results were also provided for an extension of GBAS/tdev (see [19]).

There are a few articles providing some first results on the runtime of GBAS variants of ACO: In [12, 14], GBAS with specifically chosen lower bounds for the pheromone values was studied for the OneMax problem. In those studies, it was shown that for values sufficiently close to one of an important ACO parameter called *evaporation factor*, the runtime of the algorithm until reaching the optimal solution is of order $O(n \log n)$, where n is the problem instance size, which is the same runtime behavior as that of the (1+1)-Evolutionary Algorithm (short: (1+1)-EA), a simplified GA variant that has been intensely studied in the literature. The meaning of the terms "pheromone value", and "evaporation factor" will be explained in details in Section 2.

Neumann and Witt [18] studied the behavior of a GBAS modification they called 1-ANT with specific lower *and* upper pheromone bounds, where pheromone values are re-normalized to a certain sum after each iteration. The authors showed that 1-ANT behaves identically to the (1+1)-EA on all pseudo-boolean fitness functions, provided that the evaporation factor ρ is chosen sufficiently large (basically, larger than about 1/3). Moreover, some probabilistic bounds for the behavior of 1-ANTS on OneMax for small evaporation factor were given; in particular, these bounds show that the runtime behavior is exponential for evaporation factor values close enough to zero.

In the present article, we do *not* use explicit re-normalization of pheromone values after update as described in [18], considering that an implicit re-normalization is performed in ACO anyway. (As to this point, see the results on the "scale-invariance" of ACO in Birattari et al. [1].) However, as in [21, 22, 18], upper pheromone bounds in addition to lower bounds will be used. The investigation in [14] (in the slightly changed context effected by the upper pheromone bounds) will be extended in several directions: First of all, we give a complete

picture of the behavior of the considered algorithm for the whole range of possible evaporation factor values. It will turn out that also in our framework, high evaporation factor values lead to a degeneration of the algorithm to the (1+1)-EA. In the described circumstances, however, this only happens if evaporation factor values are chosen in a small neighborhood of 1. For the main part of the range of possible evaporation factor values, the behavior of the algorithm will turn out to be essentially different from (1+1)-EA. Nevertheless, we will demonstrate that our ACO variant also shows a $O(n \log n)$ runtime behavior on OneMax for evaporation factor values where the algorithm does *not* behave identically as (1+1)-EA, provided that the evaporation factor is chosen as constant in n. This is also true for very small evaporation factor values, indicating that keeping the evaporation factor rather close to zero than close to one needs not to be a poor choice for the considered ACO variant. This observation will be confirmed at the LeadingOnes example, where even schemes letting the evaporation factor tend to zero as $n \to \infty$ can be shown to provide the $O(n^2)$ runtime behavior of (1+1)-EA.

Furthermore, we shall show that using high values of the evaporation factor has even distinct disadvantages as soon as fitness functions are considered that do not "guide" the search to a sufficient extent: For the NH and the NH-OneMax problem, we will demonstrate that the performance of both (1+1)-EA and the ACO with high evaporation factor is very poor. In particular, we are able to show that (1+1)-EA has a lower bound for the expected runtime of (exponential) order $(n/2)^{\log_2 n}$ on NH-OneMax, whereas a suitable scheme for ACO where the evaporation factor is chosen of order n^{-3} ensures an upper bound for the expected runtime of (polynomial) order $n^4 \log n$.

The plan of the paper is as follows: Section 2 introduces basic definitions and presents the investigated algorithm in formal terms. Section 3 provides three basic lemmas that will be used to prove our main results. In section 4, we provide some general results concerning the optimization of pseudo-boolean functions by ACO. Section 5 specializes the discussion to the OneMax problem and derives runtime results for two different variants of pheromone reinforcement. In section 6, we deal with the NH and the NH-OneMax problem, comparing (1+1)-EA to ACO. Section 7 is devoted to the LeadingOnes problem, and section 8 gives concluding remarks.

2 The Algorithm

In this section, we will start by describing an ACO algorithm in its general form. An ACO algorithm is an iterative and stochastic procedure, containing three basic elements. The first element is a construction graph $\mathcal{G}= (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of vertices and $\mathcal{A}$ the set of oriented edges (arcs), furnished with some rules to construct paths on $\mathcal{G}$ and to make them correspond to the feasible solutions of the search space S. The second element is a probabilistic mechanism to build such paths. This mechanism is parametrized by two vectors τ and η whose elements are associated to the elements of $\mathcal{A}$, called *pheromone* vector and *visibility* vector, respectively. Contrary to the visibility vector, the pheromone vector changes along iterations by means of a recursive rule, which is the third element of an ACO algorithm. Each of these three elements is important and influences the properties of the ACO algorithm. ACO algorithms are named and classified mainly based on the specific form of the updating rule for the pheromone vector.

We will focus here on a type of ACO algorithms with a specific pheromone updating rule.

This type can alternatively be denoted either by $\mathcal{MM}\text{AS}_{bs}$ (abbreviating $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System with best-so-far pheromone update) when referring to the key algorithmic idea developed by Stützle and Hoos in [21, 22], or by GBAS/lb/ub (GBAS with lower and upper pheromone bound) when referring to the algorithmic framework in [10, 14] that has evolved from the first ACO convergence study [9]. A third possible notation relying on the classification of ACO algorithms in Dorigo and Blum [3] would be $\text{ACO}_{bs,\tau_{min},\tau_{max}}$. For the sake of simplicity, we shall refer here to the investigated algorithm usually simply by ACO.

The results provided in section 3 are general for the considered ACO algorithm, i.e., they refer to the treatment of arbitrary combinatorial optimization problems. In the subsequent sections 4–7, we will focus on the treatment of pseudo-boolean functions as objective functions.

2.1 The Construction Graph

To present the general algorithmic framework, we use the notion of a *construction graph* in the strict formal sense as given in [9]. This approach has the advantage of combining mathematical preciseness and generality with the possibility of providing easily understandable visual representations of the algorithmic procedure in applications to concrete optimization problems.

Definition 2.1. Let an instance of a combinatorial optimization problem $f(\cdot) \rightarrow \arg\max_{x \in S} f(x)$ with finite feasible set S be given. By a *construction graph* for this instance, we mean a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ together with a solution decoding function Φ and the following rules to build paths:

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

(2) Let $\mathcal{W}$ be the set of directed paths w in $\mathcal{G}$ satisfying the following conditions:

  (i) w starts at the start node of $\mathcal{G}$;

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

  (iii) the last node on w has no successor node in $\mathcal{G}$ that is not already contained in w (i.e., w cannot be prolonged without violating (ii)).

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

The objective function value (fitness value) assigned to a complete path w is always that of $\Phi(w)$, i.e., of the corresponding feasible solution x. As a consequence, we may identify paths and solutions: Each path $w \in \bar{\mathcal{W}}$ is considered as a separate solution, such that the feasible set becomes identical to $\bar{\mathcal{W}}$. For this reason, we use the symbol x also for paths in the sequel. If an edge (i, j) is contained in a path x, we shall write $(i, j) \in x$.

In general, for a given problem instance, different kinds of construction graphs can be considered (see [7, 13] for the case of pseudo-Boolean functions). A construction graph for pseudo-Boolean functions known as chain graph [13] will be illustrated in details in section 4.

Example. Let us consider an instance of the TSP, with n cities, numbered from 1 to n, and distance matrix $D = (d_{ij})$. The goal is to find a tour starting and ending at the city with number "1", visiting all cities only once, such that the sum of the distances between consecutive cities on the tour is minimized. In this case, the state space S is the set of all closed tours visiting each city exactly once; alternatively, S can be identified with the set of the $(n-1)!$ permutations of $(2, 3, \ldots, n)$, since both the initial node and the end node of the tour are pre-defined as the city with number "1". A natural construction graph is based on the complete directed graph $\mathcal{G}$ with each element of $\mathcal{V}$ representing one of the n cities. The start node corresponds to the city numbered as "1". Each path $w \in \bar{\mathcal{W}}$ is constructed by starting in the start node and connecting iteratively the current node to one of the nodes not yet visited. If all cities are visited, the path construction terminates. The function Φ associates to each path a uniquely defined feasible solution of S, obtained from the path by completing the tour, i.e., by connecting the last visited node to the start node.

2.2 The Probabilistic Mechanism

In ACO, to each arc $(i, j) \in \mathcal{A}$, a pheromone value $\tau_{ij}(t)$ is assigned, which can change with the iteration number t of the algorithm. The pheromone values are combined to a vector $\tau(t)$, each component of which corresponds to an arc of $\mathcal{A}$.[1] In each iteration of ACO, the same probabilistic mechanism is applied to generate independently N paths in the construction graph $\mathcal{G}$. The conceptual unit generating a path is called an *ant*. Serially (or, on a parallel processor, in parallel), each of the N ants constructs a path, starting the construction process in the start node of $\mathcal{G}$. Given that an ant has already constructed the partial path u with last added node i, the path is prolonged via edge (i, j) with probability p_{ij} given by

$$p_{ij} = \frac{\tau_{ij}(t)}{\sum_{r \in \mathcal{S}(u)} \tau_{ir}(t)}\,, \tag{1}$$

where only *feasible* continuation edges (i, j) for u are taken into consideration. In (1), the set of nodes r with the property that (i, r) is a feasible continuation of partial path u is denoted by $\mathcal{S}(u)$. Therein, *feasibility* of a continuation (i, r) of a current partial path u is defined by property (2) of Definition 2.1: The complete path must be an element of $\bar{\mathcal{W}}$, so the continuation obtained by appending edge (i, r) to partial path u must be such that a path $w \in \bar{\mathcal{W}}$ starting with u and traversing then edge (i, r) exists. In most part of the present paper, we restrict ourselves to the special case $\bar{\mathcal{W}} = \mathcal{W}$, which makes the feasibility check trivial. Obviously, eq. (1) defines the probability of adding a feasible continuation edge (i, j) as proportional to the corresponding actual pheromone value $\tau_{ij}(t)$.

Often, a more general rule is used in practice which replaces the one in (1) by

$$p_{ij} = \frac{(\tau_{ij}(t))^{\alpha}\,(\eta_{ij}(u))^{\beta}}{\sum_{r \in \mathcal{S}(u)} (\tau_{ir}(t))^{\alpha}\,(\eta_{ir}(u))^{\beta}}\,,$$

[1] In the classical articles on ACO, which focus mainly on routing and scheduling problems, the values $\tau_{ij}(t)$ are arranged rather in a matrix form than in the form of a vector, with i as the row number and j as the column number. For our purposes, this is less convenient, since for the construction graphs usually applied to the optimization of pseudo-boolean functions, some elements of the matrix would then be undefined. That is why we prefer to arrange the values $\tau_{ij}(t)$ in vector form, which has an additional advantage for the mathematical representation in section 3.

where α, β are non-negative real numbers and the *visibility* vector $\eta(u)$ (which is allowed to depend on the current partial path u, but not on the iteration number t) also appears. For example, for the TSP we can choose $\eta_{ij}(u) = g(d_{ij})$, where the function g is non-increasing and has positive values. For the sake of simplicity, in the following sections we will not consider the visibility, i.e., we will set $\alpha = 1$ and $\beta = 0$.

2.3 The Pheromone Update

The algorithm is usually initialized by setting each $\tau_{ij}(0)$ to an identical initial value $\tau^{(0)}$. For some problems, this produces a uniform distribution on the set S of solutions in the first iteration, but in general this needs not to be the case.

The principle of pheromone update in ACO is that a certain share $\rho \in [0, 1]$ of pheromone "evaporates" on each edge, which is compensated by increasing pheromone as a "reward" on the edges of the "best" found path. We say then that this path has been *reinforced* in the current iteration. The parameter ρ is called the *evaporation factor*. A rather general form of the pheromone update rule is the following:

$$\tau_{ij}(m+1) = \psi((1-\rho(m)) \cdot \tau_{ij}(m) + \rho(m) \cdot I((i,j) \in \hat{x}(m))) \quad \text{for all } (i,j) \in \mathcal{A}, \tag{2}$$

where I denotes the indicator function, a general time varying evaporation factor is considered, ψ is a non-decreasing function which assumes positive values, and $\hat{x}(m)$ denotes a "best" path in some sense of the word "best". In the ACO literature, different notions of "best" have led to different types of ACO variants (cf. [7]). Two prominent variants consist in either taking the best path among those generated by the N ants in the current iteration m (*iteration-best* reinforcement), resp. in taking the best path among all paths generated by some ant in some of the iterations $1, \ldots, m$ (*best-so-far* reinforcement). In the second case, the best path is replaced as soon as some ant generates a path with a strictly higher objective function value than that of the current one.

The $\mathcal{MM}$AS$_{bs}$ algorithm considered in the following uses a best-so-far reinforcement rule:

$$\tau_{ij}(m+1) = \psi((1-\rho) \cdot \tau_{ij}(m) + \rho \cdot I((i,j) \in \hat{x}(m))) \quad \text{for all } (i,j) \in \mathcal{A}\,, \tag{3}$$

where the evaporation factor is constant along time, the path $\hat{x}(m)$ is the best seen so far in one of the previous iterations including the current one, and the function ψ is defined by

$$\psi(\tau) = \begin{cases} \tau_{min}, & \text{if } 0 \le \tau < \tau_{min}, \\ \tau, & \text{if } \tau_{min} \le \tau \le \tau_{max}, \\ \tau_{max}, & \text{if } \tau > \tau_{max}, \end{cases} \tag{4}$$

with lower and upper pheromone bounds τ_{min} and τ_{max}, respectively. We always assume $0 < \tau_{min} < \tau_{max}$.

More frequently, in $\mathcal{MM}$AS, the reward for reinforced arcs is not chosen simply as a constant ρ, but as a number $\rho \cdot c(\hat{x}(m))$ that is allowed to depend on $\hat{x}(m)$ (usually in a fitness-proportional manner). In this case, we apply

$$\tau_{ij}(m+1) = \psi\left([1 - \rho \cdot c(\hat{x}(m))] \cdot \tau_{ij}(m) + \rho \cdot c(\hat{x}(m)) \cdot I((i,j) \in \hat{x}(m))\right) \quad \text{for all } (i,j) \in \mathcal{A} \tag{5}$$

instead of (3). The function $c(\cdot)$ must be chosen in such a way that $1 - \rho \cdot c(x)$ is always nonnegative. We shall call $c(\cdot)$ the *reward function*.

In this paper, we restrict ourselves to the special case $N = 1$ of one single ant. Our derivations can be generalized to the case of $N > 1$, but providing them for $N = 1$ gives a more transparent picture.

3 General Lemmas

For being able to provide rigorous proofs of the results in the later sections, we will derive now three lemmas. The proofs of these lemmas can be found in the Appendix. We will use the lemmas for analytical investigations on the runtime of the considered type of ACO algorithms. However, they may possibly be useful also for the analysis of other meta-heuristic algorithms. In order to better understand the link between these general lemmas and the considered ACO algorithm, we first describe a relevant property of the last. After that, we will state two very general lemmas. Then, the application of the two lemmas to our ACO framework together with a more specific third lemma will be outlined.

Definition 3.1. For an arbitrary fitness function, assume that the elements $f_1, \ldots, f_M$ of the finite set $\{f(x) \,|\, x \in S\}$ are sorted in such a way that $f_1 < f_2 < \ldots < f_M$. Then we call

$$\mathcal{L}_j = \{x \in S \mid f(x) = f_j\} \quad (j = 1, \ldots, M) \tag{6}$$

the *level set* with index j $(1 \leq j \leq M)$. The number M of level sets can be as small as one, as in the case of the constant function, and as large as the cardinality of S, as in the case where there are no pairs of solutions in S with the same value of the objective function.

We now consider the stochastic process $X^m = (\hat{X}(m), \tau(m))$, $m = 0, 1, \ldots$, where $\hat{X}(m)$ is the best-so-far solution in iteration m and $\tau(m)$ is the vector of pheromone values in iteration m. We assume that the evaporation factor ρ, the initial value $\tau^{(0)}$ of the pheromone and the lower and upper pheromone bounds τ_{min} and τ_{max} are *rational* numbers. This assumption is not at all restrictive from a practical point of view. In fact, ACO algorithms are run on digital computers where real numbers are approximated by rational numbers. Since the set Q of rational numbers is closed under addition, subtraction, multiplication and division, it follows that under the considered assumption all pheromone values assumed during the process evolution are rational. The set Q is countably infinite, and therefore the realizations of X^m at any time m belong to the same countably infinite state space $S \times Q^{|\mathcal{A}|}$. A partition of the state space $S \times Q^{|\mathcal{A}|}$ into sets satisfying the conditions of the following Lemmas 3.1. and 3.2 is given by

$$S \times Q^{|\mathcal{A}|} = \bigcup_{k=1}^{M} \mathcal{L}_k \times Q^{|\mathcal{A}|} = \bigcup_{k=1}^{M} (\mathcal{L}_k \times Q^{|\mathcal{A}|}) = \bigcup_{k=1}^{M} \mathcal{X}_k \,,$$

where $\mathcal{L}_k$ is the level set with index k, and $\mathcal{X}_k = \mathcal{L}_k \times Q^{|\mathcal{A}|}$.

It is easy to see that X^m, $m = 0, 1, \ldots$ is a homogeneous Markov process. Note that if at iteration m, the best-so-far solution $\hat{x}(m)$ belongs to $\mathcal{L}_j$ for some $j \in \{1, \ldots, M-1\}$, then $\hat{x}(m')$ remains identical to $\hat{x}(m)$ in the subsequent iterations $m' = m + 1, \ldots, \bar{m} - 1$, until at some iteration $\bar{m}$ some solution $x(\bar{m}) \in \mathcal{L}_k$ with $k > j$ is found; in the last case, $\hat{x}(\bar{m}) = x(\bar{m}) \in \mathcal{L}_k$,

and so on. Therefore, at any iteration m, the vector $\hat{x}(m) \in \mathcal{L}_j$ is allowed to have transitions only towards points of the level sets $\mathcal{L}_k$, with $k \geq j$. In terms of the process X^m, $m = 0, 1, \ldots$, this can be stated as

$$p(X^{m+1} = y | X^m = x) = 0 \qquad \forall y \in \mathcal{X}_k, \text{ and } \forall x \in \mathcal{X}_j, \text{ if } \quad k < j.$$

Furthermore, because of the lower bound on the pheromone, it is easy to see that there is a strictly positive transition probability from $\hat{x}(m) \in \mathcal{L}_j$ towards each point $x \in \mathcal{L}_k$, for any $k > j$. Formally:

$$p(X^{m+1} = y | X^m = x) > 0 \qquad \forall y \in \mathcal{X}_k, \text{ and } \forall x \in \mathcal{X}_j, \text{ if } \quad k > j. \tag{7}$$

After stating the above property, we will now present the two general lemmas. The proofs of these lemmas can be found in the Appendix.

Lemma 3.1. Let $X^0, X^1, \ldots$ denote a homogeneous Markov process on a countably infinite state space $\mathcal{X}$, partitioned into the sets $\mathcal{X}_1, \ldots, \mathcal{X}_M$, each of which is also countably infinite, such that

$$p(X^{t+1} = y | X^t = x) > 0 \qquad \forall y \in \mathcal{X}_k, \text{ and } \forall x \in \mathcal{X}_j, \text{ if } \quad k > j$$
$$p(X^{t+1} = y | X^t = x) = 0 \qquad \forall y \in \mathcal{X}_k, \text{ and } \forall x \in \mathcal{X}_j, \text{ if } \quad k < j\,,$$

and let $E[T_{x \to \mathcal{X}_M}]$ denote the expected value of the time to reach the set $\mathcal{X}_M$, starting from $x \notin \mathcal{X}_M$, that is

$$E[T_{x \to \mathcal{X}_M}] := \sum_{t=1}^{\infty} t\, p(X^t \in \mathcal{X}_M, X^s \notin \mathcal{X}_M, 1 \leq s \leq t-1 \,|\, X^0 = x).$$

Then, for $x \in \mathcal{X}_j$, and $j = 1, \ldots, M-1$, it follows

$$E[T_{x \to \mathcal{X}_M}] \leq E[T_{x \to \bar{\mathcal{X}}_j}] + \sum_{k=j+1}^{M-1} \sum_{\ell=1}^{\infty} E[T_{x_{\ell,k} \to \mathcal{X}_M}]\; p(x \to x_{\ell,k})\,,$$

where $\bigcup_{\ell=1}^{\infty} \{x_{\ell,k}\} = \mathcal{X}_k$,

$$E[T_{x \to \bar{\mathcal{X}}_j}] := \sum_{t=1}^{\infty} t\, p(X^t \notin \mathcal{X}_j, X^s \in \mathcal{X}_j, 1 \leq s \leq t-1 \,|\, X^0 = x),$$

and

$$p(x \to x_{\ell,k}) := \sum_{t=1}^{\infty} p(X^t = x_{\ell,k}, X^s \in \mathcal{X}_j, 1 \leq s \leq t-1 \,|\, X^0 = x)\,.$$

Lemma 3.2. Under the hypotheses of Lemma 3.1, it follows that the expected value of the time T_M to reach the set $\mathcal{X}_M$, that is

$$E[T_M] := \sum_{t=1}^{\infty} t\, p(X^t \in \mathcal{X}_M, X^s \notin \mathcal{X}_M, 0 \leq s \leq t-1)$$

is bounded from above by

$$E[T_M] \leq \sum_{j=1}^{M-1} p(X^0 \in \mathcal{X}_j) \sum_{k=j}^{M-1} \sup_{y \in \mathcal{X}_k} \{E[T_{y \to \bar{\mathcal{X}}_k}]\} \leq \sum_{k=1}^{M-1} \sup_{y \in \mathcal{X}_k} \{E[T_{y \to \bar{\mathcal{X}}_k}]\}\,.$$

We now state another relevant property of the considered ACO algorithm: Let $\hat{x}(0) = \hat{y} \in \mathcal{L}_k$. Provided that $\tau_{max} < 1$, there is a time $t^\star = t^\star(k)$, independent on the specific initial pheromone vector $\tau(0) = \tilde{\tau}$ and on the influence of randomness (i.e., on the actual trajectory), such that, for all following times $t \geq t^\star$ for which $\hat{x}(t)$ still remains unchanged before leaving $\mathcal{L}_k$, the pheromone vector $\tau(t)$ also remains constant along time and equal to $\tau(t^\star) = \tau^\star(\hat{y})$ where $\tau^\star(\hat{y}) = \tau_{max}$ on arcs (i,j) belonging to $\hat{y}$, and $\tau^\star(\hat{y}) = \tau_{min}$ otherwise. In fact, if $\hat{X}^s = \hat{y} \in \mathcal{L}_k$, for $0 \leq s \leq t$, then the pheromone values $\tau_{ij}(s)$ of arcs not belonging to $\hat{y}$ will never be reinforced and will evolve in time as $\tau_{ij}(s) = \psi(\tilde{\tau}_{ij}(1-\rho c(\hat{y}))^s)$. It is easy to verify that after at most

$$t_1 = \left\lceil \frac{\log \tau_{min} - \log \tau_{max}}{\log(1-\rho c(\hat{y}))} \right\rceil \tag{8}$$

time steps, $\tilde{\tau}_{ij}(s)$ has reached the minimum value τ_{min} and will then not change anymore before the set $\mathcal{L}_k$ is left, provided that $1 - \rho \cdot c(x) < 1 \, \forall x \in S$. Since $\rho > 0$, the last inequality always holds if $\min_{x \in S} c(x) > 0$.

In a similar way, the pheromone values $\tau_{ij}(s)$ of arcs belonging to $\hat{y}$ will be reinforced at each time step and will evolve as

$$\tau_{ij}(s) = \psi\left(\tilde{\tau}_{ij}(1-\rho c(\hat{y}))^s + \rho c(\hat{y}) \sum_{k=0}^{s-1} (1-\rho c(\hat{y}))^k\right) = \psi(1-(1-\tilde{\tau}_{ij})(1-\rho c(\hat{y}))^s)\,.$$

After at most

$$t_2 = \left\lceil \frac{\log(1-\tau_{max}) - \log(1-\tau_{min})}{\log(1-\rho c(\hat{y}))} \right\rceil \tag{9}$$

time steps, $\tau_{ij}(s)$ has reached the maximum value τ_{max} and will then not change anymore before the set $\mathcal{L}_k$ is left. After the time $t_{max} = \max\{t_1, t_2\}$, the whole pheromone vector will remain unchanged until we leave $\mathcal{L}_k$.

For variable reward function $c(\cdot)$, the integers t_1 and t_2 given by (8) resp. (9) still depend on $\hat{y}$, and hence also t_{max} depends on $\hat{y}$, which we can write as $t_{max} = t_{max}(\hat{y})$. Let us define

$$t^\star(k) = \max\{t_{max}(\hat{y}) \,|\, \hat{y} \in \mathcal{L}_k\}. \tag{10}$$

Since the set $\mathcal{L}_k$ contains only finitely many elements, the integers $t^\star(k)$ are finite.

In terms of the process X^t, it holds that, conditionally to the event $X^0 = y = (\hat{y}, \tilde{\tau})$, with $y \in \mathcal{X}_k$, if the event $\{X^s \in \mathcal{X}_k, 1 \leq s \leq t\}$ happens for $t > t^\star(k)$, then, independently on y, the event $\{X^s = x^{t^\star(k)}, t^\star(k) \leq s \leq t\}$ happens with probability one. In other words, the specific state the process assumes within $\mathcal{X}_k$ does not change anymore after time $t^\star(k)$, until the set $\mathcal{X}_k$ is left.

Lemma 3.3. For the considered type of ACO algorithms and for $\tau_{max} < 1$, it holds that

$$\sup_{y \in \mathcal{X}_k} \{E[T_{y \to \bar{\mathcal{X}}_k}]\} \leq t^\star(k) + 1/\bar{p}_k, \tag{11}$$

where $t^\star(k)$ is given by (10), and

$$\bar{p}_k := \inf_{y \in \mathcal{L}_k} p(X^1 \notin \mathcal{X}_k \,|\, X^0 = (y, \tau^\star(y))) > 0 \quad (1 \leq k < M). \tag{12}$$

Proof. Let us abbreviate $t^\star(k)$ by $t^\star$ in the proof. We observe that indeed, $\bar{p}_k > 0$ because of the property (7) of the considered type of ACO algorithms. Furthermore, we have

$$\begin{aligned}
E[T_{y \to \bar{\mathcal{X}}_k}] &= \sum_{t=1}^{\infty} t\, p(X^t \notin \mathcal{X}_k, X^s \in \mathcal{X}_k, 1 \leq s \leq t-1 \,|\, X^0 = y) \\
&= \sum_{t=1}^{t^\star} t\, p(X^t \notin \mathcal{X}_k, X^s \in \mathcal{X}_k, 1 \leq s \leq t-1 \,|\, X^0 = y) \\
&+ \sum_{t=t^\star+1}^{\infty} t\, p(X^t \notin \mathcal{X}_k, X^s \in \mathcal{X}_k, 1 \leq s \leq t-1 \,|\, X^0 = y) \\
&\leq t^\star p\left(\bigcup_{t=1}^{t^\star} \{X^t \notin \mathcal{X}_k, X^s \in \mathcal{X}_k, 1 \leq s \leq t-1\} \,|\, X^0 = y\right) \\
&+ \sum_{t=t^\star+1}^{\infty} t\, p(X^t \notin \mathcal{X}_k, X^s \in \mathcal{X}_k, 1 \leq s \leq t-1 \,|\, X^0 = y) \\
&= t^\star\, p(X^{t^\star} \notin \mathcal{X}_k \,|\, X^0 = y) \\
&+ \sum_{t=t^\star+1}^{\infty} tp(X^t \notin \mathcal{X}_k, X^s \in \mathcal{X}_k, 1 \leq s \leq t-1 \,|\, X^0 = y).
\end{aligned}$$

For $t \geq t^\star + 1$,

$$\begin{aligned}
& p(X^t \notin \mathcal{X}_k, X^s \in \mathcal{X}_k, 1 \leq s \leq t-1 \,|\, X^0 = y) \\
= \;& p(X^s \in \mathcal{X}_k, 1 \leq s \leq t^\star \,|\, X^0 = y) \\
\cdot \;& p(X^t \notin \mathcal{X}_k, X^u \in \mathcal{X}_k, t^\star + 1 \leq u < t-1 \,|\, X^s \in \mathcal{X}_k, 1 \leq s \leq t^*, X^0 = y) \\
= \;& p(\mathcal{X}^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y) \cdot p(X^t \notin \mathcal{X}_k, X^u \in \mathcal{X}_k, t^\star + 1 \leq u < t-1 \,|\, X^{t^\star} \in \mathcal{X}_k, X^0 = y) \\
= \;& p(\mathcal{X}^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y) \cdot p(X^t \notin \mathcal{X}_k, X^u \in \mathcal{X}_k, t^\star + 1 \leq u < t-1 \,|\, X^{t^\star} = (\hat{y}, \tau^\star(\hat{y}))) \\
= \;& p(\mathcal{X}^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y) \cdot p(X^{t-t^\star} \notin \mathcal{X}_k, X^u \in \mathcal{X}_k, 1 \leq u < t-t^\star-1 \,|\, X^0 = (\hat{y}, \tau^\star(\hat{y}))).
\end{aligned}$$

Therefore,

$$\begin{aligned}
& \sum_{t=t^\star+1}^{\infty} tp(X^t \notin \mathcal{X}_k, X^s \in \mathcal{X}_k, 1 \leq s \leq t-1 \,|\, X^0 = y) \\
= \;& p(\mathcal{X}^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y) \cdot \sum_{t=t^\star+1}^{\infty} tp(X^{t-t^\star} \notin \mathcal{X}_k, X^u \in \mathcal{X}_k, 1 \leq u < t-t^\star-1 \,|\, X^0 = (\hat{y}, \tau^\star(\hat{y}))) \\
= \;& p(\mathcal{X}^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y) \cdot \sum_{v=1}^{\infty} (t^\star + v) \cdot p(X^v \notin \mathcal{X}_k, X^u \in \mathcal{X}_k, 1 \leq u < v-1 \,|\, X^0 = (\hat{y}, \tau^\star(\hat{y}))) \\
= \;& p(\mathcal{X}^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y) \cdot [t^\star \cdot 1 + 1/p(X^1 \notin \mathcal{X}_k \,|\, X^0 = (\hat{y}, \tau^\star(\hat{y})))] \\
\leq \;& p(\mathcal{X}^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y) \cdot (t^\star + 1/\bar{p}_k).
\end{aligned}$$

In total, we obtain:

$$E[T_{y \to \bar{\mathcal{X}}_k}] \leq [p(\mathcal{X}^{t^\star} \notin \mathcal{X}_k \,|\, X^0 = y) + p(\mathcal{X}^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y)] \cdot t^\star + p(X^{t^\star} \in \mathcal{X}_k \,|\, X^0 = y) \cdot 1/\bar{p}_k$$

$$\leq 1 \cdot t^\star + 1 \cdot 1/\bar{p}_k,$$

which proves the result. □

Finally, by using Lemmas 3.1–3.3, we obtain that the expected value of the time T_M to reach the set $\mathcal{X}_M$ is bounded from above by

$$E[T_M] \leq \sum_{k=1}^{M-1} (t^\star(k) + 1/\bar{p}_k) = \sum_{k=1}^{M-1} t^\star(k) + \sum_{k=1}^{M-1} 1/\bar{p}_k, \tag{13}$$

which we will use in the following sections.

4 Optimization of Pseudo-Boolean Functions

Since at the moment, the majority of analytical results on the runtime of evolutionary algorithms refer to test functions to be optimized that belong to the class of *pseudo-boolean* functions, we restrict our analysis in the present paper to this type of 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. All kinds of *subset selection* problems, as they frequently occur in the field of combinatorial optimization, can be interpreted as optimization problems for pseudo-boolean functions by interpreting $\{1,\ldots,n\}$ as a set of items and the binary vector $x = (x_1,\ldots,x_n)$ with $x_i \in \{0,1\}$ $(i = 1,\ldots,n)$ as the encoding of the subset $\{i \,|\, x_i = 1\} \subseteq \{1,\ldots,n\}$.

For optimizing pseudo-boolean or subset selection problems by an ACO approach, different construction graphs have been proposed (see [7, 13]). In this article, we consider the most simple of them, the *chain* construction graph introduced in [13]. It is given as follows: The node set $\mathcal{V}$ consists of the nodes $1,\ldots,n$, the nodes $-1,\ldots,-n$ and auxiliary nodes $\underline{0},\ldots,\underline{n}$. The arc set $\mathcal{A}$ consists of the arcs $(\underline{i-1}, i)$, $(\underline{i-1}, -i)$, $(i, \underline{i})$ and $(-i, \underline{i})$ $(i = 1,\ldots,n)$. Given a possible path w, the corresponding feasible solution $x \in S = \{0,1\}^n$ is obtained by means of the function $\Phi(w)$, as follows. If the $(2\,i)$-th node of w is 1 (-1), then $x_i = 1$ $(x_i = 0)$. An example is shown in Fig. 1 for the case $n = 4$. Note that only the pheromone values on the arcs $(\underline{i-1}, i)$ and $(\underline{i-1}, -i)$ are relevant: Since, after passing through node i or node $-i$, the agent can only proceed to node $\underline{i}$, the pheromone values on the arcs $(i, \underline{i})$ and $(-i, \underline{i})$ need not to be taken into account.

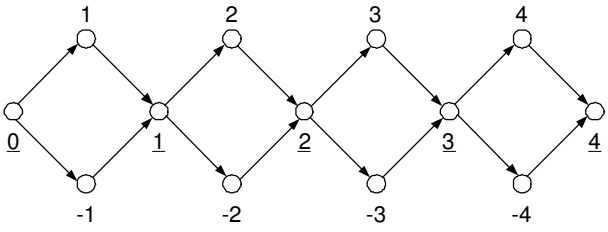

Fig. 1. Chain construction graph for $n = 4$.

For shorter notation, we shall use the following abbreviations in the sequel:

$$\tau_i^1 = \tau_{\underline{i-1},i} \quad \tau_i^0 = \tau_{\underline{i-1},-i} \quad (i = 1,\ldots,n),$$

$$p_i^1 = p_{\underline{i-1},i} \quad p_i^0 = p_{\underline{i-1},-i} \quad (i = 1, \ldots, n).$$

Thus, τ_i^1 and τ_i^0 are the pheromone values for an up-move resp. for a down-move in the i-th rhombus of the chain, and p_i^1 resp. p_i^0 are the corresponding transition probabilities. We write $\tau_i^1(m)$, $\tau_i^0(m)$, $p_i^1(m)$ resp. $p_i^0(m)$ if we refer to the state in iteration m.

With the notation above, the general pheromone update rule (5) takes the following special form for the chain graph:

$$\tau_i^k(m+1) = \psi([1-\rho\cdot c(\hat{x}(m))]\cdot\tau_i^k(m)+\rho\cdot c(\hat{x}(m))\cdot I(\hat{x}_i(m)=k)) \quad (i = 1, \ldots, n; k = 0, 1). \quad (14)$$

Throughout this paper, lower and upper pheromone bound are always chosen in a *symmetric* fashion within the interval $[0, 1]$: We choose $\tau_{min} \in (0, \frac{1}{2}]$ and set $\tau_{max} = 1 - \tau_{min}$. It can be shown that on this condition, if the sum of the pheromone values is chosen as equal to one on all links of the chain initially, then this property is preserved through all iterations:

Proposition 4.1. Let, in the general case (14), the initial pheromone values be normalized in such a way that for $m = 0$,

$$\tau_i^0(m) + \tau_i^1(m) = 1 \quad (i = 1, \ldots, n) \quad (15)$$

holds. Then (15) holds also for each $m \geq 1$.

Proof. The statement is shown by induction w.r.t. m. By assumption, it is true for $m = 0$. Assume that it is true for some $m \geq 0$. For abbreviation, set $\rho_m = \rho \cdot c(\hat{x}(m))$. From (14), we have

$$\begin{aligned}
\tau_i^1(m+1) &= \psi([1-\rho_m]\cdot\tau_i^1(m) + \rho_m \cdot I(\hat{x}_i(m)=1)) \;=\; \psi(a)\,, \\
\tau_i^0(m+1) &= \psi([1-\rho_m]\cdot\tau_i^0(m) + \rho_m \cdot I(\hat{x}_i(m)=0)) \\
&= \psi([1-\rho_m]\cdot(1-\tau_i^1(m)) + \rho_m \cdot (1 - I(\hat{x}_i(m)=1))) \;=\; \psi(1-a)\,.
\end{aligned}$$

By the graph of $\psi(x)$ versus x when $\tau_{max} = 1 - \tau_{min}$, it is evident that $\psi(a) + \psi(1-a) = 1$ $(0 \leq a \leq 1)$. Therefore, it follows $\tau_i^0(m+1) + \tau_i^1(m+1) = 1$. Hence, (15) holds for all m. □

From now on, we shall always suppose that the sum of the pheromone values is chosen as equal to one on all links initially. Since the values have to be chosen as equal on all links, this gives the initialization $\tau_i^k(0) = 1/2$ $(i = 1, \ldots, n,\ k = 0, 1)$. Then, as a consequence of Proposition 4.1, in each iteration m,

$$p_i^k(m) = \frac{\tau_i^k(m)}{\tau_i^0(m) + \tau_i^1(m)} = \tau_i^k(m) \quad (i = 1, \ldots, n;\ k = 0, 1),$$

i.e., the pheromone value $\tau_i^1(m)$ is identical to the probability $p_i^1(m)$ of having a 1-bit in position i of the solution x, and the pheromone value $\tau_i^0(m)$ is identical to the probability $p_i^0(m) = 1 - p_i^1(m)$ of having a 0-bit in position i of the solution x. In other words: Within our framework, the pheromone values can be interpreted immediately as *probabilities*. As a further consequence, we can also interpret the pheromone bounds τ_{min} and τ_{max} as probability bounds p_{min} and p_{max}, respectively. The recursions for the pheromone values (14) can be transformed to recursions for the probability values (with $p_i(m) = p_i^1(m)$):

$$p_i(m+1) = \psi([1-\rho\cdot c(\hat{x}(m))]\cdot p_i(m) + \rho\cdot c(\hat{x}(m))\cdot\hat{x}_i(m)) \quad (i = 1, \ldots, n). \quad (16)$$

(Note that $I(x_i = 1) = x_i$ for $x_i \in \{0, 1\}$.) Because of $p_i^0(m) = 1 - p_i^1(m)$, the corresponding equations for $k = 0$ (down-moves) are redundant.

Let us mention that for initial pheromone values that are *not* normalized to a sum of unity on each link, a more complicated dynamic occurs, where, however, the sums $\tau_i^1(m) + \tau_0^1(m)$ tend to 1 as m grows. In other words, a kind of "self-normalization" takes place. (For the sake of brevity, we omit the details.) Since this scaling aspect for pheromone values is not the focus of interest in the runtime analysis, but may rather be seen as a marginal phenomenon of the process, we consider it as appropriate to assume normalized pheromone values from the beginning.

The behavior of the process $(p(m)) = (p_1(m), \ldots, p_n(m))$ $(m = 0, 1, \ldots)$ is essentially influenced by the size of the evaporation factor ρ. The following proposition describes two marginal cases for the specific situation of constant reward.

Proposition 4.2. Let $c(x) = 1$ for all $x \in S$ (case of constant reward).

(a) For $\rho = 0$, all pheromone values (and therefore also all transition probabilities) remain unchanged throughout the process. In this case, the investigated ACO variant degenerates to *random search*, i.e., each configuration of S has the same probability 2^{-n}.

(b) For $\rho \geq (1 - 2p_{min})/(1 - p_{min}) = 1 - p_{min}/p_{max}$, only the two extremal probabilities are possible in each iteration:

$$p_i(m + 1) = p_{max}\, \hat{x}_i(m) + p_{min}\, (1 - \hat{x}_i(m)) \quad (i = 1, \ldots, n;\ m \geq 0).$$

Proof. (a) follows immediately from (16). As to (b), observe that $p_i(m) \in [p_{min}, p_{max}]$ for all m. Let $m \geq 0$. By (16), $p_i(m + 1) = \psi((1 - \rho)\, p_i(m) + \rho\, \hat{x}_i(m))$. If $\hat{x}_i(m) = 1$, by the monotonicity of the function ψ, we have

$$\begin{aligned} p_i(m + 1) &= \psi((1 - \rho)\, p_i(m) + \rho) \geq \psi((1 - \rho)\, p_{min} + \rho) = \psi(p_{min} + \rho(1 - p_{min})) \\ &\geq \psi(p_{min} + (1 - 2p_{min})) = \psi(1 - p_{min}) = 1 - p_{min}\,, \end{aligned}$$

so $p_i(m + 1) = 1 - p_{min} = p_{max}$. If $\hat{x}_i(m) = 0$, again by the monotonicity of ψ, it holds

$$\begin{aligned} p_i(m + 1) &= \psi((1 - \rho)\, p_i(m)) \leq \psi((1 - \rho)p_{max}) \leq \psi((1 - (1 - p_{min}/p_{max}))p_{max}) \\ &= \psi(p_{min}) = p_{min}\,, \end{aligned}$$

so $p_i(m + 1) = p_{min}$. In a compact way:

$$p_i(m + 1) = p_{max}\, \hat{x}_i(m) + p_{min}\, (1 - \hat{x}_i(m))\,.$$

We observe that in situation (b), the algorithm sets the probability of preserving a reinforced bit to $p_{max} = 1 - p_{min}$, and hence the probability of switching it to p_{min}. □

We see that for small p_{min}, the range of ρ values for which the degeneration of type (b) takes place is a rather small domain near the maximum value 1 of ρ. In the open interval $\rho \in (0, 1 - p_{min}/p_{max})$, the investigated ACO variant behaves as a "true" ACO algorithm with a potentially infinite number of pheromone values resp. transition probabilities. We note that

the degenerate ACO algorithm obtained under the premises of condition (b) in Proposition 4.2 is a generalization of a standard technique for solving problems with pseudo-boolean objective functions and studied in several articles on the runtime analysis of evolutionary algorithms: the *(1+1)-Evolutionary Algorithm* (short: (1+1)-EA). For literature references, the reader may consult [8] or [2]. In the present paper, we shall use the (1+1)-EA as the main comparison yardstick for ACO. It is given by the following procedure (see [8]):

(1+1)-EA:

1. Set $p = 1/n$.
2. Choose randomly an initial bit string $x \in \{0,1\}^n$.
3. Repeat the following mutation step: Compute x' by flipping independently each bit x_i with probability p. If $f(x')$ is better than $f(x)$, replace x by x'.

The degenerate ACO algorithm resulting under condition (b) in Proposition 4.2 is a generalization of the (1+1)-EA obtained by setting the probability p of flipping a component of x' to $p = p_{min}$.

5 Behavior of ACO on OneMax

In this section, we analyze the behavior of the investigated ACO variant on the well-known OneMax problem

$$f(x) = \sum_{i=1}^{n} x_i \tag{17}$$

$$x_i \in \{0,\, 1\} \quad (i = 1, \ldots, n).$$

It is a special problem from a broader class of single-mode optimization problems with objective function $f(x) = d(x, x^*)$ for some fixed $x^* \in \{0,\, 1\}^n$, where d denotes the Hamming distance (for OneMax, $x^* = (0, \ldots, 0)$). Since ACO with the chain construction graph shows equivalent behavior on each of these problems by symmetry, and since the same holds for (1+1)-EA, it is sufficient to investigate OneMax.

In the special case of OneMax, the level set $\mathcal{L}_j$ is just the set

$$\mathcal{L}_j = \{x \in \{0,\, 1\}^n \mid f(x) = j - 1\} \quad (j = 1, \ldots, n+1).$$

In particular, we have $M = n + 1$.

5.1 Constant Reward

First, we analyze the ACO behavior on the assumption of constant reward, i.e., for the case $c(x) = 1$ for all $x \in S = \{0,\, 1\}^n$.

Proposition 5.1. Expressed by the expected number of function evaluations until reaching the optimal solution, the expected runtime of the non-degenerate $\mathcal{MM}$AS with constant reward,

symmetric upper and lower pheromone bounds and the chain construction graph, applied to OneMax, is bounded from above by

$$n\, t^\star(p_{min}, \rho) + \frac{H_n}{p_{min}\,(1-p_{min})^{n-1}}, \tag{18}$$

where

$$t^\star(p_{min}, \rho) = \left\lceil \frac{\log p_{min} - \log(1-p_{min})}{\log(1-\rho)} \right\rceil, \tag{19}$$

and $H_n = \sum_{j=1}^{n} j^{-1}$ is the n-th harmonic number.

Proof. By (13), in general for $\mathcal{MM}$AS we have $E[T_M] \leq \sum_{k=1}^{M-1} t^\star(k) + \sum_{k=1}^{M-1} 1/\bar{p}_k$, where $t^\star(k)$ is given by (10), and $\bar{p}_k$ is given by (12). In the case of constant reward, $t^\star(k) = t^\star$ is independent on the level set index k and then

$$E[T_M] \leq (M-1)\, t^\star + \sum_{k=1}^{M-1} 1/\bar{p}_k\,.$$

Furthermore, if as in our case $\tau_{min} + \tau_{max} = 1$, it follows that

$$t^\star = t_1 = t_2 = \left\lceil \frac{\log p_{min} - \log(1-p_{min})}{\log(1-\rho)} \right\rceil .$$

For OneMax, we have

$$\begin{aligned} \bar{p}_k &= \inf_{y \in \mathcal{L}_k} p(X^1 \notin \mathcal{X}_k \,|\, X^0 = (y, \tau^\star(y))) \geq \inf_{y \in \mathcal{L}_k} p(X^1 \in \mathcal{X}_{k+1} \,|\, X^0 = (y, \tau^\star(y))) \\ &\geq \inf_{y \in \mathcal{L}_k} p(X^1 \in \mathcal{Q}(y) \,|\, X^0 = (y, \tau^\star(y)))\,, \end{aligned}$$

where $\mathcal{Q}(y) = \{z \in \mathcal{X}_{k+1} \,|\, z_i \geq y_i \;\forall i\}$. Notice that $\mathcal{Q}(y)$ is the set of points obtained from $y \in \mathcal{L}_k$ by flipping one of the $n-(k-1)$ of its 0-bits. Since $\tau = \tau^\star(y)$, the probability of a bit not to change value is equal to $p_{max} = 1 - p_{min}$, and the probability of a bit to flip is equal to p_{min}. There are $n-(k-1)$ possibilities to choose the position of the flipping bit, and the corresponding events are mutually exclusive. Therefore,

$$p(X^1 \in \mathcal{Q}(y) \,|\, X^0 = (y, \tau^\star(y))) = (n-(k-1)) \cdot p_{min} \cdot (1-p_{min})^{n-1}.$$

Hence $p_k \geq (n-(k-1)) \cdot p_{min} \cdot (1-p_{min})^{n-1}$, and

$$\begin{aligned} E[T_M] &\leq n \left\lceil \frac{\log p_{min} - \log(1-p_{min})}{\log(1-\rho)} \right\rceil + \sum_{k=1}^{n} 1/[(n-(k-1)) \cdot p_{min} \cdot (1-p_{min})^{n-1}] \\ &= n \left\lceil \frac{\log p_{min} - \log(1-p_{min})}{\log(1-\rho)} \right\rceil + \frac{1}{p_{min} \cdot (1-p_{min})^{n-1}} \sum_{j=1}^{n} 1/j\,. \end{aligned}$$

□

Special Cases: In all three special cases below, we assume $p_{min} = 1/n$ in order to preserve comparability with (1+1)-EA (cf. Proposition 4.2 (b) and the remarks after it), for which expected runtime of order $O(n \log n)$ on OneMax has been shown in the literature.

(i) $p_{min} = \frac{1}{n}$ and $\rho = 1 - \frac{1}{n-1}$. The chosen value of ρ is the lowest value for which ACO degenerates to (1+1)-EA, since for $p_{min} = 1/n$, the threshold for ρ given in Proposition 4.2 (b) is just

$$1 - p_{min}/(1 - p_{min}) = 1 - \frac{1}{n-1}.$$

In this case, we obtain

$$t^\star(p_{min}, \rho) = \left\lceil \frac{\log(1/n) - \log(1 - 1/n)}{\log(1/(n-1))} \right\rceil = \left\lceil \frac{-\log(n-1)}{-\log(n-1)} \right\rceil = 1.$$

As to be expected, one single (additional) iteration suffices here to achieve $p_j(m) = p_{max}$. We obtain

$$\frac{1}{p_{min}\,(1 - p_{min})^{n-1}} = \frac{1 - \frac{1}{n}}{\frac{1}{n}\,(1 - \frac{1}{n})^n} \sim \frac{n}{e} \quad (n \to \infty).$$

The bound (18) of Proposition 5.1 becomes asymptotically equal to

$$n + \frac{n}{e}\, H_n = O(n\, \log n).$$

Thus, in this boundary situation, the well-known $O(n \log n)$ runtime behavior of (1+1)-EA is reproduced, which must be the case in view of Proposition 4.2 (b).

(ii) $p_{min} = \frac{1}{n}$ and $\rho = \rho_0 = const$. In this case, a similar calculation as in (i) yields

$$t^\star(p_{min}, \rho) = \left\lceil \frac{\log(n-1)}{-\log(1 - \rho_0)} \right\rceil \;\leq\; \lceil c_0\, \log n \rceil$$

with $c_0 = -1/(\log(1 - \rho_0))$, such that the bound becomes

$$n\, \lceil c_0\, \log n \rceil + \frac{n}{e}\, H_n = O(n \log n).$$

(iii) $p_{min} = \frac{1}{n}$ and $\rho = \frac{1}{n}$. In this case,

$$t^\star(p_{min}, \rho) = \left\lceil \frac{\log(n-1)}{-\log(1 - \frac{1}{n})} \right\rceil \leq \left\lceil \frac{\log n}{1/n} \right\rceil = n\, \lceil \log n \rceil,$$

such that the we obtain the bound

$$n^2 \lceil \log n \rceil + \frac{n}{e}\, H_n = O(n^2\, \log n).$$

Special case (ii) shows that as long as we do not decrease ρ with growing n, the $O(n \log n)$ upper runtime bound is preserved. This is valid even for small values of ρ. To obtain the indicated favorable runtime behavior, it is not necessary to let $\rho = \rho_n$ tend to one as $n \to \infty$, as it is required for producing the (1+1)-EA boundary case. Thus, the $O(n \log n)$ behavior on OneMax holds for a "standard" range of ACO parametrizations and not only for certain extreme parameter choices where only two pheromone values can occur.

5.2 Fitness-proportional reward

In standard $\mathcal{MM}$AS implementations according to Stützle and Hoos [22], the reward function $c(\cdot)$ for reinforced solutions is not chosen as constant, but in a fitness-proportional way. Sebastiani and Torrisi [19] have shown that the general convergence property of GBAS derived in [10] also extends to the modification with fitness-proportional reward, and experiments indicated faster convergence of the modified algorithm. For this reason, we analyze in the sequel the runtime behavior of the investigated algorithm under the modified pheromone update rule, assuming

$$c(\cdot) = f(\cdot) \tag{20}$$

in this subsection. Note that (15) still holds in this context, and that the evolution of the transition probabilities is described by (16), considering (20).

In order to guarantee that the factor $1 - \rho\, c(\hat{x}(m))$ in (16) always remains non-negative, we have to impose the condition $\rho\, f(x) \leq 1$, which requires $\rho n \leq 1$ or $\rho \leq 1/n$ for the OneMax problem because of $\max f(x) = n$. We observe that the condition $1 - \rho \cdot c(x) < 1$ needed to have finite values for expressions (8), and (9) is fulfilled for all $x \neq \bar{x} = (0, \ldots, 0)$. Furthermore, for $x = \bar{x}$, from (14) we have that $\tau(m) = \tau(0)$ until the level set $\mathcal{L}_1$ is left. Hence, the time $t^\star(1)$, after which we have i.i.d. sampling until the level set $\mathcal{L}_1$ is left, is zero.

Proposition 5.2. Expressed by the expected number of function evaluations until reaching the optimal solution, the expected runtime of non-degenerate $\mathcal{MM}$AS with fitness-proportional reward, symmetric upper and lower pheromone bounds and the chain construction graph, applied to OneMax, is bounded from above by

$$\sum_{j=1}^{n-1} t^\star(p_{min}, \rho, j) + \frac{H_n}{p_{min}\,(1 - p_{min})^{n-1}}, \tag{21}$$

where

$$t^\star(p_{min}, \rho, j) = \left\lceil \frac{\log p_{min} - \log(1 - p_{min})}{\log(1 - \rho j)} \right\rceil \quad (j \geq 1), \quad t^\star(p_{min}, \rho, 0) = 0, \tag{22}$$

and H_n is the same as in Proposition 5.1.

Proof. The proof is essentially the same that of Proposition 5.1, but now $t^\star(k)$ is not independent on the level set index k. Therefore, the first summation still remains and is not equal to $n\, t^\star(p_{min}, \rho)$. □

Remark. Comparison between (19) and (22) shows that

$$t^\star(p_{min}, \rho, j) \leq t^\star(p_{min}, \rho) \quad (j = 1, \ldots, n). \tag{23}$$

As a consequence, we obtain now a *better* runtime bound than in the case of constant reward (Proposition 5.1). Note, however, that because of the necessary restriction $\rho_n \leq 1/n$ in the case of fitness-proportional reward for OneMax (cf. the beginning of this subsection), we cannot apply anymore the schemes indicated in special cases (i) and (ii) of subsection 5.1. The scheme of special case (iii) is possible. For this scheme, also under fitness-proportional reward, we do not get a runtime bound of order $O(n \log n)$ anymore, but a slightly worse behavior:

Corollary. If $p_{min} = 1/n$ and $\rho = 1/n$, then the upper runtime bound given in Proposition 5.2 is of order $O(n(\log n)^2)$.

Proof. As in the case of fixed reward, the second term in (21) is of order $O(n \log n)$. For the made choice on p_{min} and for $0 < j < n$,

$$t^\star(p_{min}, \rho, j) = \left\lceil \frac{\log(n-1)}{-\log(1-j/n)} \right\rceil. \tag{24}$$

Since $-\log(1-j/n) > j/n > 0$ for $0 < j/n < 1$, we have

$$\sum_{j=1}^{n-1} \frac{1}{-\log(1-j/n)} \; < \; \sum_{j=1}^{n-1} \frac{1}{j/n} \; = \; n\, H_{n-1} = O(n \log n).$$

It is easy to see that the order of the last bound is sharp. Considering (21), (24) and the $O(n \log n)$ behavior of the second term in (21), the assertion is obtained. □

We see that the bound has improved compared to the $O(n^2 \log n)$ bound obtained for special case (iii) in subsection 5.1. Nonetheless, a comparison with the special cases (i) and (ii) may suggest that it is disadvantageous anyway to work with evaporation factors ρ_n tending to zero with speed $O(1/n)$ as $n \to \infty$, such that fitness-proportional reward (requiring a scheme like this) does not help. This is indeed true for the "pure" OneMax case. However, we shall show in the next section that ρ_n schemes tending quickly to zero as $n \to \infty$ can be advantageous in more complex optimization problems where a OneMax-type component only forms one special aspect of the overall fitness.

6 Needle-in-a-Haystack Problem and Combination with OneMax

Contrary to prior theoretical results [12, 14, 18] that might be interpreted as arguments for the use of high evaporation factor values ρ in variants of ACO with best-so-far reinforcement, we will demonstrate in this section that applying high values of ρ to instances of some optimization problems runs the risk of making the search too "aggressive", with the possible effect of a *very* poor performance. This effect can be avoided by letting ρ tend to zero as the problem size n grows. Simultaneously, our investigation will, for the first time, reveal an essential advantage of an ACO algorithm compared to the simpler evolutionary algorithm (1+1)-EA by providing a test problem where the runtime of (1+1)-EA has an exponential lower-bound, whereas that of a type of $\mathcal{MM}$AS ACO has a polynomial upper bound.

6.1 Needle-in-a-Haystack

The disadvantage of the (1+1)-EA lies in the vulnerability with respect to lacking "guidance" the fitness function may give to the search process. Whereas high guidance is provided by OneMax (this can also be quantified by measures as fitness-distance-correlation [15]), guidance is missing or even misleading already in some very basic test problems, the simplest of which is possibly the *Needle-in-a-Haystack* (NH) problem (see, e.g., [16]) which assigns indifferent fitness values to all solutions except to the optimal solution, and gives thus no guidance for the search at all. Formally, the objective function of the NH problem is

$$f(x) = \begin{cases} 1, & \text{if } x = x^*, \\ 0, & \text{otherwise,} \end{cases}$$

where $x^* \in \{0,1\}^n$ is the optimal solution, and $x \in \{0,1\}^n$. Both for (1+1)-EA and for ACO with the chain construction graph, it can be assumed w.l.o.g. that $x^* = (1,\ldots,1)$.

The following observation shows that on NH, the required runtime of (1+1)-EA grows extremely fast with the problem instance size. This has already been proved in [8], but we repeat the (easy) argument here for the sake of completeness.

Proposition 6.1. On NH, the (1+1)-EA has an expected runtime with lower bound of order $(n/2)^n$.

Proof. For this problem, we have only two level sets, and $M = 2$. (1+1)-EA starts with a random initial solution $x^0 \in \{0,1\}^n$. After that, all bits are flipped independently from each other with probability $1/n$. None of the new solutions is accepted, unless if it is identical to $x^* = (1,\ldots,1)$. Therefore, we have i.i.d. sampling until we hit $x^\star$. Hence, the expected runtime starting from $x^0 \neq x^\star$ is the reciprocal value of the probability that x^* is reached already in a single iteration, which gives

$$E[T_{x^0 \to \mathcal{L}_M}] = \frac{1}{(1/n)^{d(x^0,x^*)} \cdot (1-1/n)^{n-d(x^0,x^*)}} = n^{d(x^0,x^*)} \cdot \left(\frac{n}{n-1}\right)^{n-d(x^0,x^*)} \geq n^{d(x^0,x^*)},$$

where d represents again the Hamming distance. Denoting the current solution in iteration t by $\hat{X}^t$, we have

$$\begin{aligned}
E[T_M] &= \sum_{t=1}^{\infty} t\, p(\hat{X}^t \in \mathcal{L}_M, \hat{X}^s \notin \mathcal{L}_M, 0 \leq s \leq t-1) \\
&\geq \sum_{t=1}^{\infty} t\, p(\hat{X}^t \in \mathcal{L}_M, \hat{X}^s \notin \mathcal{L}_M, 1 \leq s \leq t-1, \hat{X}^0 = x^0 \notin \mathcal{L}_M) \\
&= p(\hat{X}^0 = x^0 \notin \mathcal{L}_M) \sum_{t=1}^{\infty} t\, p(\hat{X}^t \in \mathcal{L}_M, \hat{X}^s \notin \mathcal{L}_M, 1 \leq s \leq t-1 | \hat{X}^0 = x^0 \notin \mathcal{L}_M) \\
&= p(\hat{X}^0 = x^0 \notin \mathcal{L}_M)\, E[T_{x^0 \to \mathcal{L}_M}]
\end{aligned}$$

If the random initial solution is $x^0 = (0,\ldots,0)$, we have $d(x^0,x^*) = n$. In any case, $p(\hat{X}^0 = x^0 \notin \mathcal{L}_M) = (1/2)^n$.

Therefore,

$$E[T_M] \geq \left(\frac{1}{2}\right)^n \cdot n^n = \left(\frac{n}{2}\right)^n .$$

□

It should be mentioned that even *random search* behaves much better than (1+1)-EA on NH: Since in a single trial of random search, the probability of hitting x^* is $(1/2)^n$, the expected runtime is 2^n.

It is easy to see that ACO with arbitrary p_{min}, arbitrary ρ and *fitness-proportional* reward performs random search on NH, which results again in an expected runtime of order 2^n. (Note that at the beginning, all p_i values are equal to $1/2$, because all pheromone values are identical initially.) This behavior becomes worse if a constant $c > 0$ is added to the objective function $f(x)$, such that it must be stated that the comparably good behavior for $c = 0$ is only a chance effect.

Concerning ACO with *fixed* reward, the following result shows that for a suitable parametrization where ρ decreases with n, ACO is at least distinctly faster on NH than (1+1)-EA, although still slower than random search. We choose $p_{min} = 1/n$ to have a fair comparison with standard $(1+1)$-EA (identical behavior in the boundary case of ρ close to one by Proposition 4.2 (b)).

Proposition 6.2. The expected runtime of the considered ACO variant with fixed reward, $p_{min} = 1/n$ and $\rho_n = 5^{-n}$ on NH has an upper bound of order $O(5^n)$.

Proof. Let us consider the time $\bar{t}_n$ until when a generic never reinforced component i takes for the last time a value not smaller than $1/4$. Accordingly to (16), we look for the largest m such that it holds

$$p_i(m) = (1-\rho)^m \cdot \frac{1}{2} \;\geq\; \frac{1}{4}\,.$$

This happens for m equal to

$$\bar{t}_n \;=\; \left\lfloor \frac{\log 2}{-\log(1-\rho)} \right\rfloor = \left\lfloor \frac{\log 2}{-\log(1-(1/5)^n)} \right\rfloor .$$

Similarly to what has been done in the proof of Lemma 3.3, one can show that

$$\begin{aligned}
E[T_M] \;&\leq\; \bar{t}_n + \sum_{\hat{y}\notin\mathcal{L}_M} p(X^{\bar{t}_n} = y) \sum_{t=1}^{\infty} t\, p(X^t \in \mathcal{X}_M, X^s \notin \mathcal{X}_M, 1 \leq s \leq t-1 \,|\, X^0 = y) \\
&=\; \bar{t}_n + \sum_{\hat{y}\notin\mathcal{L}_M} p(X^{\bar{t}_n} = y) \sum_{t=1}^{\infty} t\, p(X^t \in \mathcal{X}_M \,|\, X^{t-1} = x^{t-1}) \prod_{m=1}^{t-1} p(X^m = x^m \,|\, X^{m-1} = x^{m-1}) \\
&=\; \bar{t}_n + \sum_{\hat{y}\notin\mathcal{L}_M} p(X^{\bar{t}_n} = y) \sum_{t=1}^{\infty} t\, p(X^t \in \mathcal{X}_M \,|\, X^{t-1} = x^{t-1}) \\
&\quad\cdot \prod_{m=1}^{t-1} (1 - p(X^m \in \mathcal{X}_M \,|\, X^{m-1} = x^{m-1}))
\end{aligned}$$

where y abbreviates the expression $(\hat{y}, \psi(\tau(0)[1-\rho]^{\bar{t}_n}))$, and x^m abbreviates the expression $(\hat{y}, \psi(\tau(0)[1-\rho]^{m+\bar{t}_n}))$.

Now, we have $p(X^m \in \mathcal{X}_M \,|\, X^{m-1} = x^{m-1}) \geq p_{min}^n = n^{-n}$. As shown in the appendix by Lemma 6.1, this implies that

$$\sum_{t=1}^{\infty} t\, p(X^t \in \mathcal{X}_M \,|\, X^{t-1} = x^{t-1}) \prod_{m=1}^{t-1} (1 - p(X^m \in \mathcal{X}_M \,|\, X^{m-1} = x^{m-1})) \;\leq 1/p_{min}^n = n^n\,.$$

Therefore, since the probability of not hitting $x^\star$ during the first $\bar{t}_n$ iterations is smaller or equal to $[1-(1/4)^n]^{\bar{t}_n}$, it follows that

$$E[T_M] \leq \bar{t}_n + \sum_{\hat{y}\notin\mathcal{L}_M} p(X^{\bar{t}_n} = y)\, n^n = \bar{t}_n + p(X^{\bar{t}_n} \notin \mathcal{X}_M)\, n^n \leq \bar{t}_n + [1-(1/4)^n]^{\bar{t}_n}\, n^n\,.$$

It is easy to see that for $0 < z < 1/2$,

$$z \;\leq\; -\log(1-z) \;\leq\; (2\log 2)\cdot z. \tag{25}$$

Applying this to $z = 5^{-n}$, we obtain

$$5^n \cdot \log 2 \;\geq\; \frac{\log 2}{-\log(1-(1/5)^n)} \;\geq\; \frac{5^n}{2}\,. \tag{26}$$

As a consequence, it follows

$$\begin{aligned}\log([1-(1/4)^n]^{\bar{t}_n}\, n^n) \;&=\; \bar{t}_n\, \log[1-(1/4)^n] + n\log n \leq \frac{5^n}{2}\cdot \log\left(1-(1/4)^n\right) + n\log n \\ &\leq\; -\frac{5^n}{2}\cdot\left(\frac{1}{4}\right)^n + n\log n = -\frac{1}{2}\cdot\left(\frac{5}{4}\right)^n + n\log n \;\to -\infty, \end{aligned} \tag{27}$$

such that the expression $[1-(1/4)^n]^{\bar{t}_n}\, n^n$ must be bounded in n. On the other hand, again by (26), the term $\bar{t}_n$ is of order $O(5^n)$.

We can then conclude that the upper bound $\bar{t}_n + [1-(1/4)^n]^{\bar{t}_n}\, n^n$ for the expected runtime is of order $O(5^n)$. □

In the proof of Proposition 6.2, only rather loose estimations were used, such that the bound can possibly be improved. Moreover, it should be emphasized that long strings that can only be optimized by trial-and-error (this corresponds to the NH situation) are not typical for applications of evolutionary algorithms. Rather than that, the string to be optimized may contain a certain, comparably small part for which only trial-and-error works (no guidance by the fitness function), whereas for a larger part, the fitness function provides guidance. In the next subsection, we give an analytical result for a test function of this type.

6.2 Combining NH with OneMax

To give a precise example for a test problem of the type outlined at the end of subsection 6.1, we define the problem *NH-OneMax*, which combines the features of NH and OneMax, as follows:

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

$$f(x) \;=\; \left(\prod_{i=1}^{k} x_i\right) \cdot \left(\sum_{i=k+1}^{n} x_i + 1\right)\,,$$

$$x_i \in \{0,\, 1\} \quad (i = 1, \ldots, n).$$

As it can be seen, for obtaining fitness values larger than zero, the first factor in the definition of $f(x)$ has to be given a value 1, which is only possible if all variables $x_1, \ldots, x_k$ are set to 1. This corresponds to the solution of a NH problem. If this is achieved, the second factor in the definition of $f(x)$ has to be maximized. This corresponds to the solution of a OneMax problem. Evidently, (i) the position of the "NH bits", (ii) the values the "NH bits" have to take in order to solve the NH part of problem, and (iii) the values of the "OneMax bits" that optimize the OneMax part of the problem, can also be chosen in another way, leading to equivalent runtime behavior of ACO, and the same holds for (1+1)-EA. Thus, the example above is to be considered as a representative of a more general problem class where for some part of a string, the exact "key" has to be found by trial-and-error, and for another part, the degree of congruence with an optimal sequence has to be maximized.

To make the problem relevant, one should assume that the number of bits k for which the correct combination has to be found by trial-and-error is small compared to n (otherwise, all heuristic algorithms will have to resort essentially to blind search). The choice $k = \lceil \log_2 n \rceil$ gives a meaningful ratio. Moreover, for the sake of simplicity, we shall assume that n is a power of k, i.e., $n = 2^k$ and $k = \log_2 n$.

Proposition 6.3. The expected runtime of (1+1)-EA on NH-OneMax with $k = \log_2 n$ has lower bound $(n/2)^{\log_2 n}$.

Proof. For this problem, T_M is the smallest t such that $X^t = (1, \ldots, 1)$. Therefore, the time $T_{\bar{\mathcal{L}}_1}$ when $X^t \in \bar{\mathcal{L}}_1 = \{x \,|\, x_1 = \ldots = x_k = 1\}$ for the first time cannot be larger than T_M, i.e. $T_M \geq T_{\bar{\mathcal{L}}_1}$, and $E[T_M] \geq E[T_{\bar{\mathcal{L}}_1}]$. Let us consider any starting point $x^0 \in \mathcal{L}' := \{x \,|\, x_i = 0\,, i = 1, \ldots, k\} \subset \mathcal{L}_1$. As done in the proof of Proposition 6.1, one can show that

$$E[T_{\bar{\mathcal{L}}_1}] \geq p(\hat{X}^0 \in \mathcal{L}')\, E[T_{x^0 \to \bar{\mathcal{L}}_1}]\,,$$

where x^0 is any point in $\mathcal{L}'$, and $T_{x^0 \to \bar{\mathcal{L}}_1}$ is the time when the first level set $\mathcal{L}_1$ is left starting from x^0. We note that $p(\hat{X}^0 \in \mathcal{L}') = (1/2)^k$. As for the NH problem, when starting from $x^0 \in \mathcal{L}'$, $\hat{X}^t$ is never changed from x^0 until the first k bits are all flipped to 1. Therefore, the expected time to leave level set $\mathcal{L}_1$ starting from x^0 is the reciprocal value of the probability $(1/n)^k$ that all k bits are flipped in a single iteration, which gives $E[T_{x^0 \to \bar{\mathcal{L}}_1}] = n^k$. Hence, we have

$$E[T_M] \geq E[T_{\bar{\mathcal{L}}_1}] \geq \left(\frac{1}{2}\right)^k \cdot n^k\,.$$

As a conclusion, the expected number of steps until the NH-OneMax problem has been solved has lower bound

$$\left(\frac{1}{2}\right)^k \cdot n^k = \left(\frac{n}{2}\right)^k = \left(\frac{n}{2}\right)^{\log_2 n}.$$

□

It is easy to see that the lower bound indicated in Proposition 6.2 is not polynomial. Contrary to that, we show that ACO can cope with the problem within a polynomial expected runtime. As in sections 5 and 6.1, we choose $p_{min} = 1/n$ for the sake of comparability with (1+1)-EA.

Proposition 6.4. The expected runtime of the considered ACO variant with $p_{min} = 1/n$ and $\rho_n = n^{-3}$ on NH-OneMax with $k = \log_2 n$ has an upper bound of order $O(n^4 \log n)$.

Proof. For this problem, we have $M = n - k + 2$ level sets. We apply here Lemma 3.2, i.e.

$$E[T_M] \leq \sum_{i=1}^{n-k+1} \sup_{y \,\in\, \mathcal{X}_i} \{E[T_{y \to \bar{\mathcal{X}}_i}]\}\,.$$

We note that $T_{y \to \bar{\mathcal{X}}_1}$ is the smallest time such that the first k bits of $\hat{X}^t$ are all 1 starting from an initial point $y = (\hat{y}, \tau^0)$ with at least one 0 among the first k bits of $\hat{y}$. This means that $T_{y \to \bar{\mathcal{X}}_1}$ is the first time that the NH problem on the subvector of the first k components is solved, when starting with one 0 in the subvector. Therefore, analogously to what has been done to prove Proposition 6.2, it can be shown that

$$\sup_{y \,\in\, \mathcal{X}_1} \{E[T_{y \to \bar{\mathcal{X}}_1}]\} \leq \bar{t}_n + [1 - (1/4)^k]^{\bar{t}_n}\, n^k\,,$$

where now

$$\bar{t}_n \;=\; \left\lfloor \frac{\log 2}{-\log(1-n^{-3})} \right\rfloor ,$$

with

$$n^3 \cdot \log 2 \;\geq\; \bar{t}_n \;\geq\; \frac{n^3}{2} .$$

It also holds

$$\begin{aligned}
\log([1-(1/4)^k]^{\,\bar{t}_n}\, n^k) &= \bar{t}_n \,\log[1-(1/4)^k] + k \log n \\
&\leq (n^3/2)\log[1-(1/4)^k] + \log_2 n \,\log n \\
&\leq -(n^3/2)\,(1/4)^k + \log_2 n \,\log n \\
&= -(n^3/2)/n^2 + \log_2 n \,\log n \\
&= -n/2 + \log_2 n \,\log n \to -\infty .
\end{aligned}$$

Hence, $\bar{t}_n + [1-(1/4)^k]^{\,\bar{t}_n}\, n^k = O(n^3)$.

Analogously to what has been done to prove Proposition 5.1, we have

$$\begin{aligned}
\sum_{i=2}^{n-k+1} \sup_{y\,\in\,\mathcal{X}_i} \{E[T_{y\to\bar{\mathcal{X}}_i}]\} &\leq (n-k)t^\star(p_{min},\rho) + \sum_{i=2}^{n-k+1} 1/\bar{p}_i \\
&= (n-k)t^\star(p_{min},\rho) + \sum_{i=2}^{n-k+1} 1/[(n-k-(i-2))\cdot p_{min}\cdot(1-p_{min})^{n-1}] \\
&= (n-k)t^\star(p_{min},\rho) + \frac{1}{p_{min}(1-p_{min})^{n-1}} \sum_{j=1}^{n-k} 1/j \\
&\leq nt^\star(p_{min},\rho) + \frac{1}{p_{min}(1-p_{min})^{n-1}}\, H_n \\
&= n\left\lceil \frac{\log p_{min} - \log(1-p_{min})}{\log(1-\rho)} \right\rceil + \frac{1}{p_{min}\cdot(1-p_{min})^{n-1}} H_n \\
&\leq n\left\lceil \frac{-\log(n-1)}{\log(1-n^{-3})} \right\rceil + n\,H_n/e \\
&\leq n\left\lceil n^3 \log n \right\rceil + n\,H_n/e .
\end{aligned}$$

Finally, we get

$$E[T_M] \leq \bar{t}_n + [1-(1/4)^k]^{\,\bar{t}_n}\, n^k + n\left\lceil n^3 \log n \right\rceil + n\,H_n/e = O(n^4 \log n) .$$

Thus, all in all, the expected runtime has a bound of the claimed order. □

7 LeadingOnes

In this section, we consider the LeadingOnes function, given by

$$f(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_j \tag{28}$$

$$x_i \in \{0,\, 1\} \quad (i = 1, \ldots, n).$$

If the value $f(x)$ is returned on a generic binary string x, this means that in the first $f(x)$ positions of the string there are 1-bits, while in the next position there is a 0-bit. It is known that the expected runtime of (1+1)-EA on this function is of order $O(n^2)$. We show that an upper bound of the same order is also valid for a broad range of ACO parametrizations, including cases where both p_{min} and ρ tend to zero as $n \to \infty$. We restrict ourselves to the case of constant reward. As for OneMax, the level sets are given by

$$L_j = \{x \in \{0,\, 1\}^n \mid f(x) = j - 1\} \quad (j = 1, \ldots, n + 1).$$

Proposition 7.1. Expressed by the expected number of function evaluations until reaching the optimal solution, the expected runtime of the non-degenerate $\mathcal{MM}$AS with constant reward, symmetric upper and lower pheromone bounds with $p_{min} = 1/n$, and the chain construction graph, applied to LeadingOnes, is bounded from above by

$$n\, t^\star(p_{min}, \rho) + en^2, \tag{29}$$

where $t^\star(p_{min}, \rho)$ is given by (19). In particular, for $\rho = \rho_n$ of order $\Omega((\log n)/n)$, the expected runtime is of order $O(n^2)$. (Note that the symbol $\Omega(h(n))$ denotes a lower bound of the form constant$\cdot h(n)$.)

Proof. The proof combines that of Proposition 5.1 with the standard derivation of the $O(n^2)$ behavior of (1+1)-EA on LeadingOnes, so the presentation can be short. As in the proof of Proposition 5.1, it follows that $E[T_M] \leq n\, t^\star(p_{min}, \rho) + \sum_{k=1}^{M-1} 1/\bar{p}_k$. Moreover, in this case we have

$$p(X^1 \notin \mathcal{X}_k \,|\, X^0 = (\hat{y}, \tau^\star(\hat{y}))) = p_{min} \cdot (1 - p_{min})^k\,,$$

where $\hat{y} \in \mathcal{L}_k$. Hence $p_k = p_{min} \cdot (1 - p_{min})^k$. Since $p_{min} = 1/n$, it follows

$$p_k = \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^k \geq \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^n \geq \frac{1}{e} \cdot \frac{1}{n}\,.$$

Hence,

$$n\, t^\star(p_{min}, \rho) + en^2$$

is an upper bound for the expected runtime. The second part of the proof follows immediately by insertion of $p_{min} = 1/n$ and $\rho \geq const \cdot (\log n)/n$ into (19), considering the first inequality of (25). □

We see that for LeadingOnes, already in the case of constant reward, there exist ρ_n schemes tending to zero as $n \to \infty$ that preserve the $O(n^2)$ runtime behavior of (1+1)-EA. Presumably, it is easy to show that if the LeadingOnes problem is combined with NH analogously as OneMax with NH in subsection 6.2, ACO with a diminishing ρ_n scheme yields again considerably better results than (1+1)-EA or ACO with high ρ values.

8 Conclusions

We have analyzed the runtime complexity of an ACO variant that can be classified as $\mathcal{MM}$AS with best-so-far reinforcement on some well-known basic test problems as OneMax, Needle-in-a-Haystack and LeadingOnes. The obtained results are twofold: First, we have shown that the runtime complexity results for these test problems known from the EA literature can be transferred to ACO algorithms not only for parameter choices where ACO imitates the behavior of (1+1)-EA, but also for parametrizations where a broad range of pheromone values is used. In particular, we demonstrated that the $O(n \log n)$ result for OneMax and the $O(n^2)$ result for LeadingOnes also extend to the "true" ACO case.

Secondly, our aim was to shed light on the choice of a crucial ACO parameter, the evaporation factor ρ. Former analytical results may lead to the conjecture that in ACO algorithms working with best-so-far reinforcement, high values of ρ are generally more efficient than low values. Our results contradict this conclusion: Not only is it possible to achieve a good runtime complexity on standard test functions as OneMax or LeadingOnes also by using ρ values near to zero, but there is also a risk connected with high ρ values: It turns out that for the efficiency obtained on "easy" functions by choosing a large ρ, the price has to be paid that the more aggressive search forced in this way makes the algorithm less robust with respect to eventual "missing guidance" given by the fitness function. We were able to illustrate this effect by a combination of a OneMax with a Needle-in-a-Haystack problem where ACO with a scheme of ρ values suitably decreasing in n shows polynomial runtime behavior, whereas both (1+1)-EA and ACO with high ρ show an exponential behavior. Our results indicate that to ensure robustness, it may be advisable to keep ρ rather small and even to decrease it with growing problem instance size, but also to avoid *too* small values which would impair the performance on easy problems (or on easy parts of a problem). Of course, this tradeoff deserves future investigations.

Results for (1+1)-EA as those presented in Section 3 have already been used in other publications, although without including an explicit formal proof or a reference to it. Furthermore, for (1+1)-EA, the involved Markov process is on a finite state space. In the ACO context, the presence of the pheromone vector has to be taken into account. This leads to the case of an infinite state space. We have therefore put these assertions on a solid mathematical base by proving them explicitly in a Markov process framework that fits with the ACO situation. Hopefully, the lemmas derived in Section 3 will also be useful in a more general context.

Although the purpose of this article is not yet to provide analytical comparisons between different metaheuristics, the simultaneous consideration of an algorithm from the EA field with an ACO algorithm might prepare and stimulate investigations of this type. The most urgent goal for the next future, however, is to extend available runtime results to a broader class of test problems, especially also to NP-hard problems, for which no results seem to be available at present in the ACO literature.

Acknowledgment. The work of Walter J. Gutjahr was supported by a grant of the International short-term mobility program for scientists/researchers from Italian and foreign institutions (year 2006), awarded by the National Research Council (Consiglio Nazionale delle Ricerche), Rome, Italy.

References

[1] Birattari, M., Pellegrini, P., Dorigo, M., "On the invariance of ant colony optimization", Technical Report TR/IRIDIA/2006-025 (2006).

[2] 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 (2003).

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

[4] 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 (1999).

[5] Dorigo, M., Maniezzo, V., Colorni, A., "The Ant System: An autocatalytic optimization process", Technical Report 91–016, Dept. of Electronics, Politecnico di Milano, Italy (1991).

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

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

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

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

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

[11] Gutjahr, W.J., "A generalized convergence result for the Graph–based Ant System", *Probability in the Engineering and Informational Sciences* 17 (2003), pp. 545–569.

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

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

[14] Gutjahr, W.J., "First steps to the runtime complexity analysis of ant colony optimization", Technical Report, Dept. of Statistics and Decision Support Systems, University of Vienna, 2006, accepted for publication in: *Computers and Operations Research.*

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

[16] 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, Heidelberg, New York, pp. 174–206 (1998).

[17] Merkle, D., Middendorf, M., "Modeling the Dynamics of Ant Colony Optimization", *Evolutionary Computation* 10 (2002), pp. 235–262.

[18] Neumann, F., Witt, C., "Runtime analysis of a simple ant colony optimization algorithm", Technical Report CI-200/06, Dept. of Computer Science, University of Dortmund, Germany (2006).

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

[20] Stützle, T., Dorigo, M. "A short convergence proof for a class of ACO algorithms", *IEEE Trans. Evol. Comput.* 6 (2002), pp. 358–365.

[21] Stützle, T., Hoos, H.H., "The 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) (1997), pp. 309–314.

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

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

Appendix

Proof of Lemma 3.1
Denoting by Ω_j^q the event $\Omega_j^q = \{X^q \in \mathcal{X}_j, X^s \notin \mathcal{X}_j, 1 \le s \le q-1\}$, we have for $x \in \mathcal{X}_j$:

$$\begin{aligned} E[T_{x\to\mathcal{X}_M}] &= \sum_{t=1}^{\infty} t\, p(\Omega_M^t \,|\, X^0 = x) \\ &= \sum_{t=2}^{\infty} t \sum_{t'=1}^{t-1} \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} p(\Omega_M^t \bigcap \{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x) \\ &+ \sum_{t=1}^{\infty} t\, p(\{X^t \in \mathcal{X}_M\} \bigcap_{s=1}^{t-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x)\,. \end{aligned}$$

Denoting by B_x the first term of the last expression, we obtain

$$\begin{aligned} B_x &= \sum_{k=j+1}^{M-1} \sum_{t=2}^{\infty} \sum_{t'=1}^{t-1} \sum_{n=1}^{\infty} t\, p(\Omega_M^t \bigcap \{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x) \\ &= \sum_{k=j+1}^{M-1} \sup_{T,N\in\mathbb{N}} \left\{ \sum_{t=2}^{T} \sum_{t'=1}^{t-1} \sum_{n=1}^{N} t\, p(\Omega_M^t \bigcap \{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x) \right\} \\ &= \sum_{k=j+1}^{M-1} \sup_{T,N\in\mathbb{N}} \left\{ \sum_{n=1}^{N} \sum_{t=2}^{T} \sum_{t'=1}^{t-1} t\, p(\Omega_M^t \bigcap \{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x) \right\} \\ &= \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} \sum_{t=2}^{\infty} \sum_{t'=1}^{t-1} t\, p(\Omega_M^t \bigcap \{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x)\,, \end{aligned}$$

where we used the known result that the sum (finite or infinite) of a non-negative series is the supremum of the set of all its partial summations. By using the same results, we further obtain:

$$\begin{aligned} B_x &= \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} \sup_{T\in\mathbb{N}} \left\{ \sum_{t=2}^{T} \sum_{t'=1}^{t-1} t\, p(\Omega_M^t \bigcap \{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x) \right\} \\ &= \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} \sup_{T,T'\in\mathbb{N}} \left\{ \sum_{t'=1}^{T'} \sum_{t=t'+1}^{t'+1+T} t\, p(\Omega_M^t \bigcap \{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x) \right\} \\ &= \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} \sum_{t'=1}^{\infty} \sum_{t=t'+1}^{\infty} t\, p(\Omega_M^t \bigcap \{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} |\, X^0 = x) \\ &= \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} \sum_{t'=1}^{\infty} \sum_{t=t'+1}^{\infty} (t' + t - t')\, p\left(\{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} \,\middle|\, X^0 = x \right) \\ &\cdot \quad p(X^t \in \mathcal{X}_M, X^s \notin \mathcal{X}_M, t' < s < t | X^{t'} = x_n \in \mathcal{X}_k) \\ &= \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} \sum_{t'=1}^{\infty} t'\, p\left(\{X^{t'} = x_n \in \mathcal{X}_k\} \bigcap_{s=1}^{t'-1} \{X^s \in \mathcal{X}_j\} \,\middle|\, X^0 = x \right) \\ &\cdot \quad \sum_{t=1}^{\infty} p(X^t \in \mathcal{X}_M, X^s \notin \mathcal{X}_M, 0 < s < t | X^0 = x_n \in \mathcal{X}_k) \end{aligned}$$

$$
\begin{aligned}
&+ \sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty}\sum_{t'=1}^{\infty} p\left(\{X^{t'}=x_n\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)\\
&\cdot \sum_{t=t'+1}^{\infty}(t-t')\,p(X^t\in\mathcal{X}_M, X^s\notin\mathcal{X}_M, t'<s<t|X^{t'}=x_n\in\mathcal{X}_k)\\
&= \sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty}\sum_{t'=1}^{\infty} t'\,p\left(\{X^{t'}=x_n\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)\\
&\cdot p\left(\bigcup_{t=1}^{\infty}\left\{X^t\in\mathcal{X}_M, X^s\notin\mathcal{X}_M, 0<s<t\right\}|X^0=x_n\in\mathcal{X}_k\right)\\
&+ \sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty}\sum_{t'=1}^{\infty} p\left(\{X^{t'}=x_n\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)\\
&\cdot \sum_{t=1}^{\infty} t\,p(X^t\in\mathcal{X}_M, X^s\notin\mathcal{X}_M, 0<s<t|X^0=x_n\in\mathcal{X}_k)\\
&\le \sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty}\sum_{t'=1}^{\infty} t'\,p\left(\{X^{t'}=x_n\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)\\
&+ \sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty}\sum_{t'=1}^{\infty} p\left(\{X^{t'}=x_n\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)E[T_{x_{n,k}\to\mathcal{X}_M}]\\
&= \sum_{t'=1}^{\infty} t'\sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty} p\left(\{X^{t'}=x_n\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)\\
&+ \sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty}\sum_{t'=1}^{\infty} p\left(\{X^{t'}=x_n\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)E[T_{x_{n,k}\to\mathcal{X}_M}]\\
&= \sum_{t'=1}^{\infty} t'\sum_{k=j+1}^{M-1} p\left(\{X^{t'}\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)\\
&+ \sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty} p(x\to x_{n,k})\,E[T_{x_{n,k}\to\mathcal{X}_M}]\,.
\end{aligned}
$$

Therefore, we have

$$
\begin{aligned}
E[T_{x\to\mathcal{X}_M}] &= B_x + \sum_{t=1}^{\infty} t\,p(\{X^t\in\mathcal{X}_M\}\bigcap_{s=1}^{t-1}\{X^s\in\mathcal{X}_j\}|\,X^0=x)\\
&\le \sum_{t'=1}^{\infty} t'\sum_{k=j+1}^{M-1} p\left(\{X^{t'}\in\mathcal{X}_k\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)\\
&+ \sum_{k=j+1}^{M-1}\sum_{n=1}^{\infty} p(x\to x_{n,k})\,E[T_{x_{n,k}\to\mathcal{X}_M}]\\
&+ \sum_{t=1}^{\infty} t\,p(\{X^t\in\mathcal{X}_M\}\bigcap_{s=1}^{t-1}\{X^s\in\mathcal{X}_j\}|\,X^0=x)\\
&= \sum_{t'=1}^{\infty} t'\,p\left(\{X^{t'}\notin\mathcal{X}_j\}\bigcap_{s=1}^{t'-1}\{X^s\in\mathcal{X}_j\}\,\middle|\,X^0=x\right)
\end{aligned}
$$

$$
\begin{aligned}
&+ \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} p(x \to x_{n,k})\, E[T_{x_{n,k} \to \mathcal{X}_M}] \\
&= E[T_{x \to \bar{\mathcal{X}}_j}] + \sum_{k=j+1}^{M-1} \sum_{n=1}^{\infty} p(x \to x_{n,k})\, E[T_{x_{n,k} \to \mathcal{X}_M}] \,.
\end{aligned}
$$

□

Proof of Lemma 3.2
By the result used in the proof of Lemma 3.1, the considered quantity $E[T_M]$ can be expressed as

$$
\begin{aligned}
E[T_M] &= \sum_{t=1}^{\infty} t \sum_{j=1}^{M-1} \sum_{n=1}^{\infty} p(\Omega_M^t | X^0 = x_{n,j} \in \mathcal{X}_j)\, p(X^0 = x_{n,j} \in \mathcal{X}_j) \\
&= \sum_{j=1}^{M-1} \sum_{n=1}^{\infty} p(X^0 = x_{n,j} \in \mathcal{X}_j) \sum_{t=1}^{\infty} t\, p(\Omega_M^t | X^0 = x_{n,j} \in \mathcal{X}_j) \\
&= \sum_{j=1}^{M-1} \sum_{n=1}^{\infty} p(X^0 = x_{n,j} \in \mathcal{X}_j) E[T_{x_{n,j} \to \mathcal{X}_M}] \,.
\end{aligned}
$$

By Lemma 3.1, we have

$$
E[T_{x \to \mathcal{X}_M}] \;\leq\; E[T_{x \to \bar{\mathcal{X}}_j}] + \sum_{k=j+1}^{M-1} \sum_{m=1}^{\infty} E[T_{x_{m,k} \to \mathcal{X}_M}]\; p(x \to x_{m,k}) \,,
$$

with $x \in \mathcal{X}_j$, $x_{m,k} \in \mathcal{X}_k$ and $j = 1, \ldots, M-1$. Now, by induction, we can show that

$$
E[T_{x \to \mathcal{X}_M}] \leq \sum_{k=j}^{M-1} \sup_{y \in \mathcal{X}_k} \{E[T_{y \to \bar{\mathcal{X}}_k}]\} \,,
$$

for $x \in \mathcal{X}_j$ and $1 \leq j \leq M-1$. In fact, for $x \in \mathcal{X}_{M-1}$, it holds

$$
E[T_{x \to \mathcal{X}_M}] = E[T_{x \to \bar{\mathcal{X}}_{M-1}}] \leq \sup_{y \in \mathcal{X}_{M-1}} \{E[T_{y \to \bar{\mathcal{X}}_{M-1}}]\} \,.
$$

Let us assume now that for $x \in \mathcal{X}_j$, and $j > j'$ it holds

$$
E[T_{x \to \mathcal{X}_M}] \leq \sum_{k=j}^{M-1} \sup_{y \in \mathcal{X}_k} \{E[T_{y \to \bar{\mathcal{X}}_k}]\} \,.
$$

Then, for $x \in \mathcal{X}_{j'}$ the recursion above implies that

$$
\begin{aligned}
E[T_{x \to \mathcal{X}_M}] &\leq \sup_{y \in \mathcal{X}_{j'}} \{E[T_{y \to \bar{\mathcal{X}}_{j'}}]\} + \sum_{k=j'+1}^{M-1} \sum_{m=1}^{\infty} E[T_{x_{m,k} \to \mathcal{X}_M}]\; p(x \to x_{m,k}) \\
&\leq \sup_{y \in \mathcal{X}_{j'}} \{E[T_{y \to \bar{\mathcal{X}}_{j'}}]\} + \sum_{k=j'+1}^{M-1} \sum_{m=1}^{\infty} \sum_{s=k}^{M-1} \sup_{y \in \mathcal{X}_s} \{E[T_{y \to \bar{\mathcal{X}}_s}]\}\; p(x \to x_{m,k})
\end{aligned}
$$

$$
\begin{aligned}
&\leq \sup_{y\in\mathcal{X}_{j'}}\{E[T_{y\to\bar{\mathcal{X}}_{j'}}]\} + \sum_{k=j'+1}^{M-1}\sum_{m=1}^{\infty}\sum_{s=j'+1}^{M-1}\sup_{y\in\mathcal{X}_s}\{E[T_{y\to\bar{\mathcal{X}}_s}]\}\; p(x\to x_{m,k})\\
&\leq \sup_{y\in\mathcal{X}_{j'}}\{E[T_{y\to\bar{\mathcal{X}}_{j'}}]\} + \sum_{s=j'+1}^{M-1}\sup_{y\in\mathcal{X}_s}\{E[T_{y\to\bar{\mathcal{X}}_s}]\}\sum_{k=j'+1}^{M}\sum_{m=1}^{\infty}\; p(x\to x_{m,k})\\
&= \sup_{y\in\mathcal{X}_{j'}}\{E[T_{y\to\bar{\mathcal{X}}_{j'}}]\} + \sum_{s=j'+1}^{M-1}\sup_{y\in\mathcal{X}_s}\{E[T_{y\to\bar{\mathcal{X}}_s}]\}\, p(x\to\bar{\mathcal{X}}_{j'})\\
&\leq \sum_{j=j'}^{M-1}\sup_{y\in\mathcal{X}_j}\{E[T_{y\to\bar{\mathcal{X}}_j}]\}\,.
\end{aligned}
$$

Therefore, the result is also true for $j = j'$. Hence, the result is proved for all $1 \leq j \leq M-1$.

By using this result in the third expression of the block at the beginning of the proof, we obtain

$$
\begin{aligned}
E[T_M] &= \sum_{j=1}^{M-1}\sum_{n=1}^{\infty} p(X^0 = x_{n,j}\in\mathcal{X}_j)E[T_{x_{n,j}\to\mathcal{X}_M}]\\
&\leq \sum_{j=1}^{M-1}\sum_{n=1}^{\infty} p(X^0 = x_{n,j}\in\mathcal{X}_j)\sum_{k=j}^{M-1}\sup_{y\in\mathcal{X}_k}\{E[T_{y\to\bar{\mathcal{X}}_k}]\}\\
&= \sum_{j=1}^{M-1}\sum_{k=j}^{M-1}\sup_{y\in\mathcal{X}_k}\{E[T_{y\to\bar{\mathcal{X}}_k}]\}\sum_{n=1}^{\infty} p(X^0 = x_{n,j}\in\mathcal{X}_j)\\
&= \sum_{j=1}^{M-1}\sum_{k=j}^{M-1}\sup_{y\in\mathcal{X}_k}\{E[T_{y\to\bar{\mathcal{X}}_k}]\}p(X^0\in\mathcal{X}_j)\\
&\leq \sum_{k=1}^{M-1}\sup_{y\in\mathcal{X}_k}\{E[T_{y\to\bar{\mathcal{X}}_k}]\}\sum_{j=1}^{M} p(X^0\in\mathcal{X}_j)\\
&= \sum_{k=1}^{M-1}\sup_{y\in\mathcal{X}_k}\{E[T_{y\to\bar{\mathcal{X}}_k}]\}\,.
\end{aligned}
$$

□

Lemma 6.1. Let us consider a temporal sequence of independent Bernoulli events $\mathcal{E}_t$, $t = 1, 2, \ldots$ with $p_t = p(\mathcal{E}_t = 1) \geq p > 0$. Let T be the first time such that $\mathcal{E}_t = 1$ along the time sequence $t = 1, 2, \ldots$. Then, it follows

$$
E[T] := \sum_{t=1}^{\infty} t\, p_t \prod_{i=1}^{t-1}(1-p_i) \;\leq \sum_{t=1}^{\infty} t\, p\,(1-p)^{t-1} = 1/p\,.
$$

Proof. It is well-known that for two random variables $X_1 \geq 0$ and $X_2 \geq 0$ with existing expected values and distribution functions F_1 and F_2, respectively, the stochastic dominance property $F_1(t) \geq F_2(t)\ \forall t$ implies $E[X_1] \leq E[X_2]$. Now, let $\mathcal{E}_t$ be as defined, and let $\mathcal{E}'_t$ be a sequence of independent Bernoulli events with $p(\mathcal{E}'_t = 1) = p\ \forall t$. Then the corresponding distribution functions for the first success times T of $(\mathcal{E}_t)$ and T' of $(\mathcal{E}'_t)$ are

$$
F_1(t) = p(T \leq t) = 1 - \prod_{i=1}^{t}(1-p_i)
$$

resp.

$$F_2(t) = p(T' \leq t) = 1 - (1-p)^t \leq F_1(t),$$

which shows $E[T] \leq E[T'] = 1/p$. □