First Steps to the Runtime Complexity Analysis of Ant Colony Optimization

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

Abstract: The paper presents results on the runtime complexity of two ant colony optimization (ACO) algorithms: Ant System, the oldest ACO variant, and GBAS, the first ACO variant for which theoretical convergence results have been established. In both cases, as the class of test problems under consideration, a slight generalization of the well-known OneMax test function has been chosen. The techniques used for the runtime analysis of the two algorithms differ: In the case of GBAS, the expected runtime until the optimal solution is reached is studied by a direct bound estimation approach inspired by comparable results for the (1+1) evolutionary algorithm (EA). A runtime bound of order $O(m \log m)$, where m is the problem instance size, is obtained. In the case of Ant System, the original discrete stochastic process is approximated by a suitable continuous deterministic process. The validity of the approximation is shown by means of a rigid convergence theorem exploiting a classical result from mathematical learning theory. Using this approximation, it is demonstrated that for the considered OneMax-type problems, a runtime of order $O(m \log(1/\epsilon))$ until reaching an expected *relative* solution quality of $1 - \epsilon$, and a runtime of $O(m \log m)$ until reaching the *optimal* solution with high probability can be predicted. Our results are the first to show competitiveness in runtime complexity with (1+1) EA on OneMax for a proper ACO algorithm.

1 Introduction

Ant algorithms applied to the heuristic solution of combinatorial optimization problems, an approach that has originally been developed by Dorigo, Maniezzo and Colorni [6, 7] in the early 1990s, have evolved to a powerful, versatile optimization tool with a still rapidly growing number of publications and numerous applications in diverse areas of operations research, management science and technology. Under the name "Ant Colony Optimization" (ACO) coined by Dorigo and Di Caro [4], this approach has become one of the best-known metaheuristic techniques, side to side with tabu search, genetic and evolutionary algorithms, simulated annealing, iterated local search, variable neighborhood search and some other prominent general-purpose search methods.

As Dorigo and Blum point out in their recent survey on *theory* of ACO [3], after a huge amount of experimental work on ACO carried out in the recent years, the time has come to deepen the understanding of ACO by an increased effort to extend its theoretical foundations. In the meantime, ACO theory can be considered as well-equipped with *convergence* results (see Gutjahr [11, 12, 13], Stützle and Dorigo [24], Sebastiani and Torrisi [23]), and also some first investigations on the finite-time dynamics of ACO are already available (see Merkle and Middendorf [18], Gutjahr [17]). However, none of these articles addresses the important question how the (expected) runtime until a solution of a given quality is obtained scales in the size of the problem instance. This is the traditional runtime complexity problem which has been a focus of interest in the field of analysis of algorithms since

the very beginnings of theoretical computer science. For certain *evolutionary algorithms* (EAs), a considerable progress concerning runtime complexity analysis has been achieved in the last years. In particular, for the (1+1)-EA, there is already a comparably large family of optimization problems for which statements on the expected runtime until reaching the optimal solution have been given (see, e.g., Droste, Jansen and Wegener [9], Neumann and Wegener [19], or Wegener and Witt [27]). It would be desirable to have similar results also for the ACO domain. This issue has also been formulated in [3] as the current *Open Problem 1* of ACO theory: to prove convergence speed results for ACO algorithms that resemble in their spirit those already obtained in the EA field, starting with relatively simple test problems such as, e.g., OneMax.

The present article presents solutions to the mentioned "Open Problem 1" by giving first runtime complexity results for two genuine ACO algorithms: The one of them is *GBAS with lower pheromone bounds* (GBAS/lb), an ACO variant that has not only been subject to several theoretical investigations in the ACO literature (see [12] or [23]), but also forms the backbone of the S-ACO algorithm [14, 15] for *stochastic* combinatorial optimization, which has already proven successful in applications (see [22, 1]). The other investigated algorithm is *Ant System* (AS), the oldest and conceptually simplest ACO algorithm, from which several more sophisticated variants have been derived in later years. We use completely different techniques for the analysis of these two algorithms. For GBAS/lb, a technique borrowed from the EA literature is applied. This technique exploits a certain monotonicity structure of the algorithm and fails to work in the case of the analysis of AS. Therefore, a more intricate method using an approximation of the AS process by the solution of a system of ordinary differential equations is applied to get results on the runtime behavior in the AS case. For both algorithms, the chosen class of test problems is as a slight generalization of the well-known OneMax problem.

The organization of the paper is as follows: Section 2 describes the overall structure of the ACO algorithms under consideration. In section 3, the analyzed test problems are introduced. Section 4 contains the analysis of GBAS/lb. In section 5, the process ACDP approximating AS is described, and a convergence result connecting AS to the ACDP is given. Then, the runtime analysis for the chosen problem class is carried out. Section 6, finally, contains concluding remarks.

2 General Structure of the Investigated ACO Algorithms

We formulate the investigated algorithms without reference to *visibility values* but use only the so-called *pheromone values* (for an explanation of these notions, the reader is referred to [8]).

First of all, let us recall the following formal definition of a *construction graph* of an ACO algorithm (cf. [11, 12]): Let an instance of a combinatorial optimization problem 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 arc set $\mathcal{A}$ together with a 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 $\mathcal{C}$ satisfying the following conditions:

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

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

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

  Then Φ maps a subset $\bar{\mathcal{W}}$ of the set $\mathcal{W}$ onto the set of feasible solutions of the given problem instance. In other words: To each path w in $\bar{\mathcal{W}}$, there corresponds (via Φ) a feasible solution, and to each feasible solution, there corresponds (via Φ^{-1}) at least one path in $\bar{\mathcal{W}}$.

Both ACO algorithms investigated in this paper have the general structure indicated in Fig. 1; they differ by the special ways the steps "initialize pheromone values" and "do pheromone update" are performed.

Procedure ACO

```
Initialize pheromone values τ_kl on the arcs (k,l) of the construction graph C;
for iteration n = 1,2,... {
  for ant s = 1,...,S {
    set k, the current position of the ant, equal to the start node of C;
    set u, the current path of the ant, equal to the empty list;
    while (a feasible continuation (k,l) of the path u of the ant exists) {
      select successor node l with probability p_kl, where
        p_kl = 0, if continuation (k,l) is infeasible, and
        p_kl = τ_kl / (Σ_(k,r) τ_kr), where the sum is over all feasible (k,r), otherwise;
      continue the current path u of the ant to node l by adding arc (k,l) and setting k := l;
    }
  }
  do pheromone update;
}
```

Fig. 1. Pseudocode of ACO.

In the pseudocode given by Fig. 1, *feasibility* of a continuation (k,l) of a current (partial) path u is defined by property (2) of the definition of the construction graph: The complete path must be an element of $\bar{\mathcal{W}}$, so the continuation $u+(k,l)$ obtained by concatenating u with arc (k,l) must be such that a path $w \in \bar{\mathcal{W}}$ with prefix $u+(k,l)$ exists. The objective function value assigned to a complete path w is always that of $\Phi(w)$, i.e., of the corresponding feasible solution. In this way, paths can be considered as solutions.

3 Test Problems

We consider a class of simple pseudo-boolean functions with unique local optima. The feasible set is the set $\{0,1\}^m$ of binary vectors of length m, or, in other terms, the set of all subsets of a given collection of m items $1,\ldots,m$. A solution is represented as a vector $x=(x_1,\ldots,x_m)$ with $x_i \in \{0,1\}$ $(i=1,\ldots,m)$. Now let us define the cost of a solution x as the Hamming distance $d_H(x,x^*)$ of x to a fixed specific solution x^*, i.e., as the number of bits by which x and x^* differ. It is obvious that x^* is the only optimal solution.[1] In order to re-formulate the problem as a maximization problem, the objective function (fitness function) may be re-defined as

$$f(x) = m - d_H(x,x^*), \tag{1}$$

i.e., as the number of "correct" bits in x, or, more generally, as

$$f(x) = m + c - d_H(x,x^*) \tag{2}$$

with some constant $c \geq 0$. Note that although (1) and (2) are equivalent as optimization problems, the behavior of ACO differs for different values of c. A special case is the case $x^* = (1,\ldots,1)$; this

[1] Since $d_H(x,x^*) = ||x-x^*||_2^2$ for $x, x^* \in \{0,1\}^m$, our class of test functions can be considered as a discrete analogue to the *sphere model* used in the analysis of evolution strategies.

yields the well-known *OneMax test function* given by $f(x) = \sum_{i=1}^{m} x_i + c$, where again the addition of a constant c does not change the optimization problem itself, but may influence the behavior of the solution algorithm.

Several different construction graphs have been proposed for pseudo-boolean or subset selection problems (see [8, 17]). In this article, we restrict the analysis to the most simple of them, the *chain* construction graph introduced in [17]. It is given as follows: The node set of $\mathcal{C}$ consists of the nodes $1, \ldots, m$, the nodes $-1, \ldots, -m$ and auxiliary nodes $\underline{0}, \ldots, \underline{m}$. The arc set consists of the arcs $(\underline{i-1}, i)$, $(\underline{i-1}, -i)$, $(i, \underline{i})$ and $(-i, \underline{i})$ $(i = 1, \ldots, m)$. The choice of node i resp. node $-i$ means that $x_i = 1$ resp. $x_i = 0$. An example is shown in Fig. 2. 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 ant has only the choice to 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. We shall choose the initial pheromone value $\tau(0)$ as $1/(2m)$, i.e., as the reciprocal of the number of relevant arcs of the chain. In this paper, the length $L(w)$ of a path w is measured by the number of decision nodes on w, which means that $L(w) = m$.

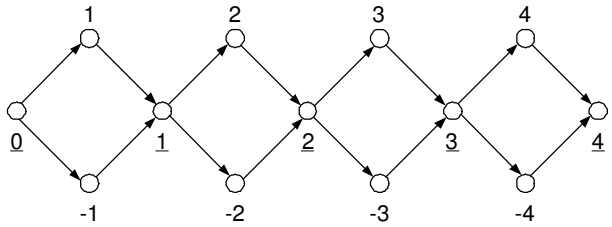

Fig. 2. Chain construction graph for $m = 4$.

4 GBAS with Lower Pheromone Bound

4.1 The Algorithm

In this section, we investigate an ACO variant with a "global-best" pheromone update rule (see Gambardella and Dorigo [10]) that has been formally described in [12] and named there GBAS/tdlb (Graph-Based Ant System with time-dependent lower pheromone bound). The idea of applying pheromone bounds in ACO is due to Stützle and Hoos [25, 26]. As the name says, in GBAS/tdlb, a (lower) bound on pheromone values is used which is allowed to depend on time, i.e., on the current iteration of the procedure. In the present paper, we do not exploit the possible time-dependency of the bound, but instead keep the bound fixed over time. Consequently, the algorithm will be called GBAS/lb instead of GBAS/tdlb in the sequel. However, we allow that the bound is chosen in dependence of the problem instance size.

Now let us specify the pheromone handling in GBAS/lb. Pheromone *initialization* is done by setting $\tau_{kl} = \tau_{kl}(0) = \tau(0)$ for all arcs (k, l) of $\mathcal{C}$, where the constant $\tau(0)$ is chosen inversely proportional to the number $|\mathcal{A}|$ of arcs of $\mathcal{C}$. Pheromone *update* consists in letting a certain share ρ of pheromone "evaporate" on all arcs, and compensating this effect afterwards by placing pheromone as a "reward" on the arcs of the best found path. We say then that this path has been *reinforced* in the current iteration. More specifically, pheromone update in iteration n is performed as follows:

$$\tau_{kl}(n+1) := \begin{cases} \max\left((1-\rho)\,\tau_{kl}(n) + \rho \,/\, L(\hat{w}(n)),\ \tau_{min}\right), & \text{if } (k,l) \in \hat{w}(n), \\ \max\left((1-\rho)\,\tau_{kl}(n),\ \tau_{min}\right), & \text{otherwise,} \end{cases} \tag{3}$$

where $0 < \rho < 1$, and τ_{min} is a positive number chosen in advance, the lower pheromone bound. Therein, $\hat{w}(n)$ is the best path found until the end of iteration n, which is stored in a specific (list) variable and replaced each time some ant finds a path with better objective function value. $L(w)$ is the length of path w, expressed by some suitable measure. In this paper, $L(w) = m$ as stated above. If it is clear from the context to which iteration we refer, we shall write only τ_{kl} instead of $\tau_{kl}(n)$.

In the following subsection, the number S of ants is chosen as $S = 1$, which is not an essential restriction in the case of GBAS/lb.

4.2 Complexity Analysis

It is evident that due to the symmetry properties of the *chain* construction graph and the equal initial pheromone values, the behavior of GBAS/lb is equivalent for each specific choice of x^*. Thus, the analysis of the class of test problems (2) can be restricted to the analysis of OneMax without loss of generality, as long as the *chain* is used. (This does not hold anymore for certain other construction graphs, e.g., the *disk* introduced in [17].)

We shall write GBAS/lb/chain for the algorithm GBAS/lb when using the chain construction graph.

Based on the definitions and notation above, the question of the expected runtime until the optimal solution has been found can now be addressed. As in the literature on the runtime analysis of evolutionary algorithms (see, e.g., [9]), we measure the runtime by the number of necessary *function evaluations*. The following result shows that for a suitable parametrization of GBAS/lb/chain with ρ near to unity, the expected runtime has an upper bound of order $O(m \log m)$. The second part of the proof essentially follows the lines of the proof of the corresponding (1+1) EA result in [9].

Theorem 4.1. Expressed by the expected number of function evaluations until reaching the optimal solution, the expected runtime of GBAS/lb/chain with $\rho = 1 - a/m$ and $\tau_{min} = a/m^2$ ($a > 0$ constant), applied to one of the problems (2), is bounded above by $(2e^{2a}/a) \cdot mH_m$ for $m \geq 2a$, where $H_m = \sum_{j=1}^{m} j^{-1}$ is the mth harmonic number. This bound is of order $O(m \log m)$.

All proofs can be found in the Appendix.

Remark 1. Although the behavior of GBAS/lb with a parameter choice as indicated in Theorem 4.1 resembles that of the (1+1)-EA in some sense, it should be emphasized that the two algorithms are *not* simply *identical* [2]. In particular, it is easily seen that for an arc in the construction graph, an infinite number of different pheromone values and therefore also an infinite number of different transition probabilities is possible in GBAS/lb even when the parameter settings of Theorem 4.1 are chosen, such that our algorithm possesses the characteristic ACO property of allowing a gradual reinforcement of "good" arcs by repeated pheromone increments. This is contrary to the (1+1)-EA, where the only learned information is the knowledge of the best solution so far, and only two different probability values, namely $1 - 1/n$ for bits that are part of this best solution so far and $1/n$ for bits that are not part of it, occur during the construction of the solution.

Remark 2. As a function of a, the bound in Theorem 4.1 takes its minimum value for $a = 1/2$.

[2] The similar runtime behavior of certain simple variants of ACO and the (1+1)-EA has been observed for the first time in [16], where also a preliminary version of Theorem 4.1 has been presented. In a recent technical report [20], Neumann and Witt outline that by providing GBAS with a suitable lower *and* a suitable upper bound, by an appropriate renormalization of pheromone values, and by the choice of the evaporation parameter in a special range, the algorithm can be forced to mimic the behavior of the (1+1)-EA *exactly*. They also investigate the runtime of their algorithmic variant on OneMax for parameter choices where the behavior is different from (1+1)-EA, but do not obtain $O(m \log m)$ upper bounds for this case.

However, this does not necessarily imply that the choice $a = 1/2$ provides the best expected runtime, since Theorem 4.1 only gives an *upper* bound.

5 Ant System

5.1 AS and ACDP

The runtime of the GBAS variant of ACO is comparably easy to analyze mathematically because of its monotonicity property: the fitness of those solutions that are used for pheromone reinforcement is always increasing during the process. This is analogous to the behavior of the (1+1)-EA, where a new solution is only accepted if it is better than the current solution. Such a monotonous behavior lends itself to be exploited for the core arguments of a runtime analysis. Unfortunately, the majority of ACO variants proposed in the literature and used in practice do not show the mentioned monotonicity. Iteration-based reinforcement schemes, both in their fitness-proportional and in their rank-based version, admit the possibility that during an iteration, solutions that are worse than some other solutions seen previously are reinforced. Thus, the solution quality can deteriorate temporarily. In some sense, this is more typical for a "natural" learning process than the forced monotonicity of global-based learning[3], and experimental results show that the performance of global-best reinforcement schemes can be improved by a mix with or sometimes even by a replacement by iteration-based schemes.

The theoretical runtime analysis of ACO variants using iteration-based schemes, however, turns out to be surprisingly difficult. In this article, we present first results of the following type: As the ACO variant to be investigated, we choose the historically oldest and simplest ACO algorithm, *Ant System* (AS), developed by Dorigo, Maniezzo and Colorni in their pioneering papers [6, 7]. In order to facilitate the analysis at least to some extent, we study an approximation to the AS process for the asymptotic case of small pheromone values introduced under the name ACDP (Associated Continuous Deterministic Process) in [17], which turns out to be the "core process" governing the essential behavior of AS with small ρ, if random fluctuations are neglected. (Note that in AS, small to medium-sized pheromone values have been used in experimental research and seem to produce better results for this algorithmic variant than large values.) We shall give results on the runtime complexity of the ACDP for the test functions (2), including OneMax, and we shall interpret them in terms of AS. Instead of investigating the convergence speed of the best solution found so far (as in the previous section), we shall study in this section the speed of convergence of the pheromone, which is a stronger type of convergence. This will be discussed at the end of section 5.

For describing AS, we use again the more general ACO pseudocode of Fig. 1 and specify the steps "initialize pheromone values" and "do pheromone update" for AS. Again, *pheromone initialization* is done by $\tau_{kl}(0) = \tau(0)$ for all arcs (k, l), where the constant $\tau(0)$ can be chosen in a problem-specific manner. *Pheromone update* follows now a *fitness-proportional* rule: In the case $S = 1$ of a single ant,

$$\tau_{kl}(n+1) = (1-\rho) \cdot \tau_{kl}(n) + \rho \cdot I((k,l) \in X_n) \cdot f(X_n), \tag{4}$$

where X_n is the (random) walk traversed by the ant in round n,

$$I(\text{statement}) = \begin{cases} 1, & \text{if statement is true,} \\ 0, & \text{otherwise} \end{cases}$$

is the indicator function, and $(k, l) \in X$ means that arc (k, l) is contained in the path defined by walk X. This means that each arc lying on the path X_n traversed by the ant in the current iteration gets a pheromone update that is proportional to the fitness of X_n. It is always assumed that $f \geq 0$.

[3] In natural ant colonies, from where the ACO paradigm is derived, global-best reinforcement does not occur.

In the case of $S > 1$ ants, the contributions of the S paths traversed by the S ants are averaged:

$$\tau_{kl}(n+1) = (1-\rho) \cdot \tau_{kl}(n) + \frac{\rho}{S} \sum_{s=1}^{s} I((k,l) \in X_n^{(s)}) \cdot f(X_n^{(s)}). \tag{5}$$

For the description of the ACDP, let us start with the definition of the *passage fitness*: As in section 4, let $\mathcal{A}$ be the set of arcs (k,l) of $\mathcal{C}$. By indexing the elements of $\mathcal{A}$ in an arbitrary order, the pheromone values τ_{kl} can be comprised to a vector τ with $|\mathcal{A}|$ nonnegative components, $\tau \in [0,\infty[^{|\mathcal{A}|}$. By $\tau(n)$, we denote the vector of pheromone values $\tau_{kl}(n)$ $((k,l) \in \mathcal{A})$ in iteration n of the ACO process.

Definition 5.1. The *passage fitness* of arc (k,l) under a given pheromone vector $\tau = (\tau_{kl})$ is the random variable

$$I((k,l) \in X) \cdot f(X), \tag{6}$$

where X is the random walk of a fixed ant under pheromone values τ_{kl}. The *expected passage fitness* of (k,l) under τ is the mathematical expectation of the passage fitness, i.e., the value

$$F_{kl}(\tau) \;=\; E\left(I((k,l) \in X) \cdot f(X)\right) \;=\; \sum_{x \in S:(k,l) \in x} Pr\{X = x\} \cdot f(x) \tag{7}$$

with X as defined above.

Now let us turn to the asymptotic case where ρ is small. It is obvious that if ρ is reduced, such that pheromone changes in each round become small, a comparably larger number of rounds must be executed to get substantial changes. Therefore, in order to be able to plot the asymptotic behavior, it is 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 it is unity, i.e., we assume that $\rho = 1/M$. In this scaling, an iteration takes $dt = 1/M = \rho$ time units. Re-scaled time will be denoted by the symbol t in the sequel. It should be mentioned that this type of re-scaling for obtaining asymptotic results on "slow learning" is customary and well-established in mathematical learning theory (see Norman [21]).

Definition 5.2. The ACDP (Associated Continuous Deterministic Process) is given as the solution $\bar{\tau}(t)$ of the system

$$\frac{d\bar{\tau}_{kl}}{dt} = F_{kl}(\bar{\tau}) - \bar{\tau}_{kl} \qquad ((k,l) \in \mathcal{A}) \tag{8}$$

of ordinary differential equations, where $\bar{\tau} = (\bar{\tau}_{kl})$ is the pheromone vector at time t, and $F_{kl}(\bar{\tau})$ is the expected passage fitness of arc (k,l) under $\bar{\tau}$.

The ACDP has been introduced in [17]. However, the question in which sense the vector-valued process $\bar{\tau}(t)$ approximates the process $\tau(n)$ of pheromone vectors in AS has not yet been answered there by a mathematical convergence result; this question will be addressed in subsection 5.2 of this paper. For re-scaling the process $\bar{\tau}(t)$ back to the process $\tau(n)$, it should be observed that iteration n corresponds to time $n/M = n\rho$. Thus, $\tau(n)$ is approximated by $\bar{\tau}(n\rho)$.

As in section 2, we use the *chain* as the underlying construction graph in this section. With this choice, we shall denote the processes AS and ACDP more specifically as AS/chain and ACDP/chain, respectively.

5.2 Convergence of AS to ACDP

In order to be able to consider the ACDP as a suitable approximation to AS, such that runtime results on the ACDP can be interpreted in terms of AS, we need a mathematical convergence result showing

that in the described asymptotic, the trajectories of AS converge indeed to those of the ACDP. To set up such a convergence result, we use a general theorem proved by Norman in chapter 8 of his book [21]. We recapitulate the theorem here in a slightly specialized form in order not to overload the text with a generality that is not needed in our context. Moreover, we adapt Norman's notation to that of this paper, such that we stay consistent with the usual notation in the ACO literature. Norman defines a vector-valued parametrized stochastic process $(\tau^\rho(n))$ by the recursion

$$\tau^\rho(n+1) = (1-\rho)\tau^\rho(n) + \rho\, U(\tau^\rho(n), X_n), \tag{9}$$

where X_n is a random variable obeying a given distribution

$$p(\tau, G) = \Pr\{X_n \in G \mid \tau^\rho(n) = \tau\}.$$

The vectors $\tau^\rho(n)$ are assumed to be contained in a subset $I_0 \subseteq R^N$, and the parameter ρ is taken from a bounded set $J \in R^+$ with $\inf J = 0$. The analysis aims at the asymptotic case $\rho \to 0$, $n \to \infty$ and $n\rho \to t \in R$, which corresponds to the "intermediate term" of the process $(\tau^\rho(n))$ for small ρ. Norman introduces the normalized increment $H^\rho(n)$ as

$$H^\rho(n) = \frac{\tau^\rho(n+1) - \tau^\rho(n)}{\rho},$$

and derives the following vector, matrix and scalar from it:

$$W(\tau) = E(H^\rho(n) \mid \tau^\rho(n) = \tau),$$

$$S(\tau) = E((H^\rho(n) - W(\tau))^2 \mid \tau^\rho(n) = \tau),$$

and

$$R(\tau) = E(||H^\rho(n)||^3 \mid \tau^\rho(n) = \tau).$$

Therein, $\tau^2 = \tau\,\tau^*$ and $||\tau|| = (\tau^*\,\tau)^{1/2}$ for a column vector $\tau \in R^N$, the star symbol denoting transposition. For an $[N \times N]$ matrix A, its norm is defined as $||A|| = \left(\sum_{i=1}^N \sum_{j=1}^N a_{ij}^2\right)^{1/2}$.

Theorem 5.1 (Norman [21]). Let the following assumptions be satisfied:

(i) The vector function $W(\tau)$ is differentiable on I_0 in the sense that there exists an $[N \times N]$ matrix valued function $W'(\tau)$ on I_0 such that

$$\lim_{\sigma \to \tau, \tau \in I_0} \frac{||W(\sigma) - W(\tau) - W'(\tau)(\sigma - \tau)||}{||\sigma - \tau||} = 0.$$

(ii) $W'(\tau)$ is bounded.

(iii) Both $W'(\tau)$ and $S(\tau)$ satisfy Lipschitz conditions:

$$\sup_{\tau, \sigma \in I_0, \tau \neq \sigma} \frac{||W'(\tau) - W'(\sigma)||}{||\tau - \sigma||} < \infty,$$

and analogously for $S(\tau)$.

(iv) $R(\tau)$ is bounded.

Then, for $\mu_n(\rho, \tau) = E_\tau(\tau^\rho(n))$ and $\omega_n(\rho, \tau) = E_\tau(||\tau^\rho(n) - \mu_n(\rho, \tau)||^2)$, where E_τ denotes the expectation in a process with initial vector τ, the following holds:

(A) $\omega_n(\rho,\tau) = O(\rho)$ uniformly in $\tau \in I_0$ and $n\rho \le T$, for any $T \ge 0$.

(B) For any $\tau \in I_0$, the differential equation

$$\dot{g}(t) = W(g(t)) \tag{10}$$

has a unique solution $g(t) = g(t,\tau)$ with $g(0) = \tau$. For all $t \ge 0$, $g(t) \in I_0$, and

$$\mu_n(\rho,\tau) - g(n\rho,\tau) = O(\rho)$$

uniformly in $\tau \in I_0$ and $n\tau \le T$.

Assertion (A) of the theorem basically says that in the considered asymptotic $\rho \to 0$, $n \to \infty$ and $n\rho \to t \in [0,T]$, the random fluctuations of the vectors $\tau^\rho(n)$ around their mean values tend to zero at least with speed of order $O(\rho)$. Assertion (B) presents a way to determine the limits of the mean values (and therefore also of the random process values): Solving (10) provides us with a function g, to which the re-scaled process values converge. An approximation of $\tau^\rho(n)$ for small ρ and large n is then given by $g(t,\tau(0))$, where $\tau(0)$ is the initial vector, and $t = n\rho$. It should be observed that the convergence is uniform in each fixed interval $t \in [0,T]$.

Let us remark that Norman's original theorem also contains a third assertion (C), a central-limit theorem stating asymptotic normality of the re-scaled vectors $\tau^\rho(n)$. We omit this issue here.

Theorem 5.1 can be applied to the analysis of AS/chain for pseudoboolean functions as follows: As in section 4, we restrict ourselves to the case $S = 1$ of a single ant.[4] The vector $\tau^\rho(n)$ is the pheromone vector in iteration n for a given evaporation factor ρ. The set J from where ρ is taken is chosen as the interval $]0,1[$. The number N of components of the vector $\tau^\rho(n)$ is the number $2m$ of relevant pheromone values $\tau_i^{(k)}$ $(i = 1,\dots,n; k = 0,1)$ in the chain. $I_0 \subseteq R^{2m}$ is a domain containing all possible pheromone vectors $\tau^\rho(n)$. The function U introduced in (9), finally, is given by the vector $U(\tau^\rho(n), X_n)$ with components

$$U_{(k,l)}(\tau^\rho(n), X_n) = I((k,l) \in X_n) \cdot f(X_n),$$

where (k,l) runs through all $2m$ relevant arcs of the chain graph. Evidently, (9) reproduces then our recursion (4). Moreover,

$$H^\rho(n) = -\tau^\rho(n) + U(\tau^\rho(n), X_n) = -\tau^\rho(n) + (I((k,l) \in X_n)) \cdot f(X_n), \tag{11}$$

where $(I((k,l) \in X_n))$ denotes the vector with components $I((k,l) \in X_n)$ for each relevant arc (k,l).

As a consequence,

$$W(\tau) = -\tau + F(\tau), \tag{12}$$

where $F(\tau)$ is the vector of expected passage fitness values $F_{(k,l)}(\tau)$. Then, (10) just reproduces the defining system of differential equations (8) of the ACDP.

The following result shows that under mild conditions, ACDP/chain suitably approximates AS/chain for all pseudoboolean fitness functions:

Proposition 5.1. Let f be a pseudoboolean function with the following properties:

(a) For a suitable initial value $\tau^\rho(0)$, all vectors $\tau^\rho(n)$ of the corresponding AS/chain process stay within a compact set $I_0 \in R^{2m}$ with $\min\{\tau_i^{(0)} + \tau_i^{(1)} \mid \tau \in I_0\} > 0$ for $i = 1,\dots,m$.

[4] The case of $S > 1$ ants which can be treated analogously, with similar results, is omitted here for the sake of brevity.

(b) For each arc (k, l) of the chain, the expected passage fitness $F_{k,l}(\tau)$ is 3-times (partially) differentiable in I_0.

Then, for the process $\tau^\rho(n)$ with initial values $\tau^\rho(0)$, the convergence statements (A) and (B) of Theorem 5.1 hold for AS/chain applied to f, and the limiting process $g(t)$ is the ACDP given by (8).

Now we apply Proposition 5.1 to the special case of the functions (2).

Lemma 5.1. For the fitness functions (2), the pheromone values in AS/chain with $S = 1$ always satifsy

$$c \le \tau_i^{(0)}(n) + \tau_i^{(1)}(n) \le m + c \quad (i = 1, \ldots, m),$$

provided that these inequalities are valid for the initial pheromone values.

According to Lemma 5.1, we can define the pheromone vector domain I_0 for the functions (2) as

$$I_0 = \{\tau \in {R_+}^{2m} \mid c \le \tau_i^{(0)} + \tau_i^{(1)} \le m + c \ \ (i = 1, \ldots, m)\}. \tag{13}$$

This is a bounded and closed and hence a compact subset of ${R_+}^{2m}$. For $c > 0$, the set I_0 given by (13) satisfies condition (a) of Proposition 5.1. (Note that this is not true anymore for the boundary case $c = 0$.) Let us now turn to condition (b). It is immediately seen that

$$F_{\underline{i-1},i}(\tau) = \frac{\tau_i^{(1)}}{\tau_i^{(0)} + \tau_i^{(1)}} \cdot \left[1 + c + \sum_{j \ne i} \frac{\tau_j^{(1)}}{\tau_j^{(0)} + \tau_j^{(1)}}\right] \tag{14}$$

and

$$F_{\underline{i-1},-i}(\tau) = \frac{\tau_i^{(0)}}{\tau_i^{(0)} + \tau_i^{(1)}} \cdot \left[c + \sum_{j \ne i} \frac{\tau_j^{(1)}}{\tau_j^{(0)} + \tau_j^{(1)}}\right]. \tag{15}$$

The first factors on the r.h.s. of (14) and (15) yield the probabilities that the considered arc (up-move and down-move, respectively) in link i of the chain is chosen. The second factors yield the conditional expected fitness values, given that the considered arc has been chosen. By the independence between the links, this expected fitness can be computed by summation of the up-move probabilities over all links different from link i, plus c, plus 1 for link i in the case of (14). Thus, in the case $c > 0$, for $\tau \in I_0$ as in (13), all $F_{k,l}(\tau)$ are infinitely times differentiable functions of τ, which verifies condition (b). In total, this yields:

Proposition 5.2. For each of the problems (2) with $c > 0$, the trajectories of AS/chain converge in the sense of Theorem 5.1 to those of the ACDP/chain.

5.3 Complexity Analysis

Based on the insight of Proposition 5.2, AS/chain for (2) can be analyzed by investigating the runtime behavior of the corresponding ACDP. We still have to decide on the initial pheromone value $\tau(0)$. As discussed in [17], for OneMax on ACDP/chain, the sum $\tau_1^{(0)} + \tau_1^{(1)}$ converges to the maximum fitness value f_{max}. If initially the total pheromone flow $\tau_1^{(0)} + \tau_1^{(1)} = 2\tau(0)$ out of node $\underline{0}$ is chosen distinctly smaller or larger than f_{max}, this flow will quickly scale to the "correct" order of magnitude f_{max}. Here, we are not interested in this scaling process for the overall pheromone amount, but in the core optimization process, i.e., the identification of the optimizer x^* by the adjustment of the pheromone values *relative* to each other. Thus, in order to filter scaling effects out, it is reasonable to choose

$\tau_1^{(0)}(0)$ and $\tau_1^{(1)}(0)$, and therefore also all other pheromone values, as half of the maximum fitness, i.e., we set $\tau(0) = f_{max}/2 = (m+c)/2$ for a fitness function of type (2).

We shall express our runtime result on ACDP/chain under the indicated circumstances in terms of the relative fitness of a solution:

Definition 5.3. The *relative fitness* of a solution x is given by $(f(x) - f_{min})/(f_{max} - f_{min})$, where $f_{max} = \max\{f(x) \mid x \in \{0,1\}^m\}$ and $f_{min} = \min\{f(x) \mid x \in \{0,1\}^m\}$.

For the interpretation of the following proposition, it should be noted that although the ACDP is a deterministic process in terms of the pheromone evolution, the walk of the ant based on the pheromone vector $\tau(t)$ at time t is still random, such that it is appropriate to consider its *expected* relative fitness.

Proposition 5.3. For each of the problems (2) with $c > 0$ and for each $\epsilon > 0$, the required amount of (re-scaled) time t until the expected relative fitness of the ant's walk in the ACDP/chain reaches a value of $1-\epsilon$, is bounded from above by $2(m+c)\log((1-\epsilon)/\epsilon)$ and from below by $(m/2)\log((1-\epsilon)/\epsilon)$.

Proposition 5.3 shows that the considered runtime (in re-scaled time units) is of order $\Theta(m\log(1/\epsilon))$ for large m and small ϵ.

Remark 1. Let us now interpret Proposition 5.3 in terms of the original process AS/chain. Given that the parameters of AS are chosen in such a way that the AS process is approximated by the ACDP to a sufficient degree of accuracy, our results say that the required number of function evaluations until AS/chain, applied to one of the functions (2), reaches an expected relative fitness of $1-\epsilon$, is of order

$$\Theta\left(\frac{m}{\rho}\,\log\frac{1}{\epsilon}\right). \tag{16}$$

This follows immediately from the fact that during one unit of re-scaled time, $1/\rho$ iterations and hence the same number of function evaluations are performed. Evidently, the indicated runtime increases with decreasing ρ, such that there is a tradeoff between the accuracy of our approximation and the runtime of the algorithm: The prediction is the better, the smaller ρ is; however, one will decide to choose ρ not *too* small because this would negatively affect the computational effort.[5]

In order to ensure that the runtime of AS (as opposed to ACDP) until reaching the $1-\epsilon$ level remains within a bound of type (16) *with a pre-specified probability*, we would have to refine the mathematical analysis by taking from Norman's theorem not only the "law of large numbers" parts A – B, as cited in Theorem 5.1, but also the "central limit theorem" part C omitted here. This analysis is outside the scope of the present paper. Let us confine ourselves to the remark that for obtaining a probabilistic bound of the outlined type, ρ would probably have to be chosen in dependence of m, which would increase the runtime to $\Theta\left(\frac{m}{\rho(m)}\,\log\frac{1}{\epsilon}\right)$ with some suitably decreasing $\rho(m)$. However, this is presumably not the best thing to do from a practical point of view: The real aim is not to design the optimization procedure in such a way that it gets mathematically predictable, but to find good solutions as fast as possible. Now, the noise superimposed to the deterministic core process (as described by the ACDP) by some not too small ρ can also *reduce* the runtime until the expected relative fitness has reached the level $1-\epsilon$ for the first time, since it can effect that by chance, one of the pheromone vectors achieves this level already earlier than it is reached in the correspondingly

[5] At the first look, the mentioned tradeoff may seem more harmful than it is. Actually, it is typical for several well-established applications of probability theory. E.g., the classical statistical tests are derived from asymptotic approximations of test statistics distributions. Making the applied formulas *exact* would require to let the sample size and hence the experimental effort go to infinity.

scaled ACDP. Therefore, choosing ρ as independent of m is *not* to be expected to worsen the runtime behavior, compared to the strategy where ρ is decreased with increasing m.

Remark 2. The consideration in Remark 1 refers to the runtime until some *relative* solution quality is reached, e.g., solutions with an (expected) fitness only 1% below the maximum fitness. By setting $\epsilon = \epsilon(m) = \epsilon_0/m$ with some fixed $\epsilon_0 > 0$, we obtain the time until the distance of the expected *absolute* fitness to the maximum fitness is smaller than the pre-defined value ϵ_0. For the problems (2), the distance between the best and a second-best fitness value is unity, therefore a pheromone vector $\tau(n)$ supporting an expected absolute fitness value of $f_{max} - \epsilon_0$ $(0 < \epsilon_0 \ll 1)$ guarantees that the ant chooses the optimal solution with a probability of at least $1 - \epsilon_0 \approx 1$, since from

$$f_{max} - \epsilon_0 = \text{ expected absolute fitness } \leq f_{max} \cdot Pr\{f_{max} \text{ reached}\} + (f_{max} - 1) \cdot Pr\{f_{max} \text{ not reached}\},$$

the inequality $Pr\{f_{max} \text{ not reached}\} \leq \epsilon_0$ follows. Evidently, the re-scaled runtime until this situation occurs for ACDP/chain is of order

$$\Theta\left(m \log \frac{m}{\epsilon_0}\right) = \Theta\left(m(\log m - \log \epsilon_0)\right) = \Theta(m \log m),$$

and the corresponding runtime for AS/chain is therefore of order $\Theta((m/\rho) \log m)$. As argued in Remark 1, it is suitable to keep ρ independent of m. For constant ρ, we obtain in this way the same $\Theta(m \log m)$ runtime behavior as that for the (1+1)-EA and for the GBAS/lb parametrization investigated in section 2.

Nevertheless, it should be observed that the type of analysis in this section is basically different from that in section 4: In section 4, we asked for the expected time until a *solution* of a given quality is produced for the first time. In section 5, we were interested in the time until the process has evolved to a *pheromone vector* that supports solutions with a given quality — in other words, until the ACO algorithm has *learned* to produce high-quality solutions. Simple examples show that the time until the first hit of a good solution is usually distinctly smaller than the time needed for representing the characteristics of this solution in the pheromone values. However, in [16, 17] it is argued that for being able to deal efficiently with more complex optimization problems, the ability to learn, i.e., to create internal representations of essential properties of (parts of) good solutions, is of crucial importance. In this sense, the type of runtime properties investigated in section 3 can be considered as stronger than that of section 4.

6 Conclusions

We have analyzed the runtime complexity of two ACO algorithms, GBAS/lb and Ant System, applied to the optimization of pseudobooelean functions of OneMax type. In both cases, computation times of order $O(m \log m)$ were found as sufficient for reaching the optimum, which is the same order as that determined previously by other authors for the (1+1) evolutionary algorithm. For AS, our runtime complexity bound primarily refers to the pheromone behavior, which gives a stronger assertion than the corresponding one for the best solution found so far. For this reason, it is possible that the expected runtime until reaching the optimal solution for the first time is even of a more favorable order than $O(m \log m)$.

Admittedly, our analysis refers to rather basic test functions. However, as Proposition 5.1 shows, it is probable that the analysis can be extended to more general pseudoboolean functions, and perhaps even beyond this range to fitness functions for permutation-type problems or problems with still another combinatorial structure. In particular, the approach of section 3 provides a tool to overcome the intrinsic difficulties of the analysis of ACO variants as Ant System or Ranked-Based ACO [2]

where the reinforcement does not follow the monotonous "global-best" scheme, but instead allows temporary deteriorations of the currently reinforced solution.

A specific feature of our results is that they do not only address the expected runtime until the *optimal* solution has been found (as most papers on the runtime complexity analysis of evolutionary algorithms do), but also the expected runtime until some *pre-defined (relative) solution quality* has been reached. This may turn out as especially helpful as soon as NP-hard problems are included into the analysis: In this situation, computation times until reaching the optimum are inefficient anyway, but from a practical point of view, one would like to have answers to questions as: "How much time will it take until we get a solution quality of, say, 1% below the optimum, which is sufficiently good for our application?" It would be a very desirable justification of a metaheuristic algorithm if it could be shown that for some NP-hard problems, the expected runtime until reaching an expected relative fitness of $1-\epsilon$ in the sense of Definition 5.1 is only polynomial. (At least for certain NP-hard problems, complexity theory does not exclude this possibility.)

Obviously, many open problems and topics for future research remain. Let us list only a few of them: (1) As mentioned above, an extension of the results to other (linear and nonlinear) pseudo-boolean functions as well as to problems with permutation-structured solutions should be the next aim. (2) ACO variants such as ACS [5] , Rank-Based ACO [2] or MAX-MIN Ant System [26] should be included into the analysis, and the performance of these variants might be compared in terms of runtime complexity results. (3) The influence of parameter choices such as that of the evaporation factor ρ or the number S of ants on the runtime behavior should be investigated in detail. (4) A refined mathematical analysis might make the relations between ACO and the auxiliary process ACDP used in section 3 more explicit, e.g., by linking their runtime behavior in terms of stochastic bounds. (5) Optimization problems could possibly be identified for which ACO has a better or a worse runtime complexity than some other metaheuristic algorithm outside the ACO domain. This might lead to a classification of optimization problems according to the most suitable metaheuristic for treating them, which would provide a very helpful guideline for the choice between metaheuristic techniques in applications.

Although all the mentioned problems are mathematically rather challenging, the chances are good that several of them can be solved in the near future.

Acknowledgement. The author is indebted to an anonymous reviewer for profound and detailed comments on the first version of this paper. In particular, the proof of Proposition 5.3 has been given a more compact and comprehensible form based on these valuable comments.

References

[1] Brailsford, S.C., Gutjahr, W.J., Rauner, M., Zeppelzauer, W., "Combined discrete-event simulation and ant colony optimisation approach for selecting optimal screening policies for diabetic retinopathy", *Computational Management Science*, in print (published online: 15th Aug. 2006).

[2] 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* 7, pp. 25-38 (1999).

[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., Gambardella, L.M., "Ant Colony System: a cooperative approach to the traveling salesman problem", *IEEE Transactions on Evolutionary Computation* 1 (1997), pp. 53–66.

[6] 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).

[7] 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.

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

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

[10] Gambardella, L.M., Dorigo, M., "Ant-Q: A reinforcement learning approach to the traveling salesman problem", in: Proc. ML-95, Twelfth Intern. Conf. on Machine Learning (Morgan Kaufman, Palo Alto, CA, 1995) 252–260.

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

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

[13] 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.

[14] Gutjahr, W.J., "A converging ACO algorithm for stochastic combinatorial optimization", *Proc. SAGA 2003* (Stochastic Algorithms: Foundations and Applications), Springer LNCS 2827, pp. 10-25 (2003).

[15] Gutjahr, W.J., "S-ACO: An ant-based approach to combinatorial optimization under uncertainty", *Proc. ANTS 2004* (4th International Workshop on Ant Colony Optimization and Swarm Intelligence), Springer LNCS 3172, pp. 238-249 (2004).

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

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

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

[19] Neumann, F., Wegener, I., "Rundomized local search, evolutionary algorithms, and the minimum spanning tree ", Technical Report CI-165/04, Dept. of Computer Science, University of Dortmund, Germany (2004).

[20] 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).

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

[22] Rauner, M., Brailsford, S.C., Gutjahr, W.J., Zeppelzauer, W., "Optimal screening policies for diabetic retinopathy using a combined discrete-event simulation and ant colony optimization approach", *Proc. Int. Conference on Health Sciences Simulation, Western MultiConference '05*, eds.: Anderson, J.G., Katzper, M. (2005), pp. 147–152.

[23] 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.

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

[25] 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.

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

[27] 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 Theorem 4.1. It is easy to see that the pheromone values cannot exceed an upper bound $\tau_{max} = 1/m$: Initially, $\tau_{kl}(0) \leq 1/m$ by our choice $\tau(0) = 1/(2m)$. By induction with respect to m, it immediately follows from (3) that then also $\tau_{kl}(n) \leq 1/m$ for all n must hold. Note that on an arc (k, l) that is reinforced in every iteration, pheromone can expressed for the choice made for ρ, τ_0, τ_{min} and m, as

$$\tau_{kl}(n) = 1/m - \frac{1}{2m}\,(a/m)^n,$$

which converges to $1/m = \tau_{max}$.

We set $q = 1 - \rho = a/m$. By move 1 and move 0 on position i we denote the choice of an arc $(\underline{i-1}, i)$ and $(\underline{i-1}, -i)$, respectively. We shall also call move 1 an *up-move* and move 0 a *down-move*. Instead of $\tau_{\underline{i-1},i}$ and $\tau_{\underline{i-1},-i}$, we use the shorter notation $\tau_i^{(1)}$ and $\tau_i^{(0)}$, respectively. Analogously, we denote the probability of move k on position i by $p_i^{(k)}$ $(k = 0, 1)$. Since $p_i^{(k)} = 1/(1 + \tau_i^{(1-k)}/\tau_i^{(k)})$ and $\tau_{min} \leq \tau_i^{(k)} \leq \tau_{max}$ $(k = 0, 1)$, the minimal possible value for $p_i^{(k)}$ is

$$p_{min} = \frac{\tau_{min}}{\tau_{max} + \tau_{min}} = \frac{a}{m + a} = \frac{q}{1 + q},$$

and the maximal possible value for $p_i^{(k)}$ is

$$p_{max} = \frac{\tau_{max}}{\tau_{max} + \tau_{min}} = \frac{m}{m + a} = \frac{1}{1 + q}.$$

Without loss of generality, we assume that the problem of type (2) under consideration is just OneMax, i.e., $x^* = (1, \ldots, 1)$. Suppose move k $(k = 0, 1)$ on position i has been reinforced in the last iteration. Then, the two pheromone values for position i in the current iteration satisfy

$$\tau_i^{(k)} \geq q\tau_{min} + \frac{1 - q}{m} \geq \frac{1 - q}{m},$$

and

$$\tau_i^{(1-k)} = \tau_{min} = \frac{q}{m},$$

where the second formula follows from the fact that a move that is not reinforced makes the pheromone fall to a value not larger than $q\tau_{max} = q/m = \tau_{min}$. Hence

$$p_i^{(1-k)} = \frac{\tau_i^{(1-k)}}{\tau_i^{(k)} + \tau_i^{(1-k)}} \leq \frac{q/m}{(1-q)/m + q/m} = q$$

and therefore

$$p_i^{(k)} = 1 - p_i^{(1-k)} \geq 1 - q.$$

By A_j, we denote the set of all solutions x with $f(x) = j$. Assume that in the iteration before the current iteration, a currently best solution $y \in A_j$ has been reinforced ($0 \leq j \leq m-1$). We look for a lower bound for the probability π that the solution x generated in the current iteration differs from y only by flipping one or more 0-bits to an 1-bit, such that $x \in A_k$ with $k > j$. Obviously,

$$\pi \geq Pr\{\text{only one of the } m \text{ bits of } y \text{ is flipped from 0 to 1}\}.$$

Thus, we bound the probability π from below by only considering those events where *exactly one* of the moves corresponding to the 0-bits of y becomes an up-move. Evidently, these events are mutually exclusive. There are $m - j$ possibilities to choose the position of the desired up-move. For each of these possibilities, we have:

- The probability that at the position chosen among the $m - j$ possible positions, an up-move occurs, is larger or equal to $p_{min} = q/(1+q)$.
- The probability that all other $m - 1$ positions are not changed is larger or equal to $(1-q)^{m-1}$.

Thus,

$$\pi \geq \frac{q}{q+1} (1-q)^{m-1} (m-j).$$

Therefore, the expected runtime until some current solution $y \in A_j$ is changed to a solution in $A_{j+1} \cup \ldots \cup A_m$ (which leads to a new currently best solution) is bounded from above by

$$\frac{q+1}{q} (1-q)^{-m+1} (m-j)^{-1}.$$

In the worst case, the current solution y starts in the first iteration in the set A_0 and passes in each solution improvement event from some A_j only to the immediately next A_{j+1} , which results in an upper bound for the total expected runtime of

$$\sum_{j=0}^{m-1} \frac{q+1}{q} (1-q)^{-m+1} (m-j)^{-1} = \frac{q+1}{q} (1-q)^{-m+1} \sum_{j=1}^{m} \frac{1}{j}$$

$$= \frac{q+1}{q(1-q)^{m-1}} H_m = \frac{a+m}{a(1-a/m)^{m-1}} H_m < \frac{2}{a} \frac{1}{(1-a/m)^{m-1}} \cdot mH_m < \frac{2e^{2a}}{a} mH_m,$$

where the last inequality follows by exponentiation from $(m-1)\log(1-a/m) \geq (m-1) \cdot (-2a/m) > -2a$ for $a/m \leq 1/2$. Since $2e^{2a}/a$ is a constant and $H_m = O(\log m)$, the bound is of order $O(m \log m)$ $(m \to \infty)$. □

Proof of Proposition 5.1. It suffices to verify that the conditions (i) – (iv) of Theorem 5.1 are satisfied. Condition (a) ensures that I_0 can be taken as the vector domain I_0 in Theorem 5.1. By (12)

and condition (b), $W(\tau)$ is 3-times (partially) differentiable in I_0. In particular, the Jacobi matrix $W'(\tau)$ of $W(\tau)$ exists, satisfying the condition on the matrix $W'(\tau)$ in Theorem 5.1. This proves (i). Furthermore, since also the elements of $W'(\tau)$ are differentiable, hence in particular continuous and therefore bounded on the compact set I_0, condition (ii) is satisfied as well.

For verifying the first part of condition (iii), it is sufficient to show that each element of the matrix $W'(\tau)$ is Lipschitz. This, however, follows from the fact that the gradient of each element of $W'(\tau)$ is differentiable itself, hence continuous and therefore bounded on I_0.

For the second part of (iii), we consider the component in line a and column b of the matrix $S(\tau)$,

$$E((H_a^\rho(n) - W_a(\tau))\,(H_b^\rho(n) - W_b(\tau)) \mid \tau^\rho(n) = \tau),$$

where a and b correspond to *arcs* of the chain graph. Short computation using (11) shows that the expression above is equal to

$$\begin{aligned} &E(I(a \in X_n)\, I(b \in X_n)\, (f(X_n))^2 \mid \tau^\rho(n) = \tau) - F_a(\tau)F_b(\tau) \\ &= \sum_{x \in \{0,1\}^m} \Pr\{X_n = x \mid \tau^\rho(n) = \tau\} \cdot I(a \in x \text{ and } b \in x)\, (f(x))^2 - F_a(\tau)F_b(\tau). \end{aligned} \tag{17}$$

By condition (b), $F_a(\tau)$ and $F_b(\tau)$ are 3-times differentiable on I_0. On the other hand, since

$$\Pr\{X_n = x \mid \tau^\rho(n) = \tau\} = \prod_{i=1}^{m} \frac{\tau_i^{(x_i)}}{\tau_i^{(0)} + \tau_i^{(1)}},$$

where x_i is the ith bit of x, and $\tau_i^{(0)} + \tau_i^{(1)} > 0$ by condition (a), also the probability $\Pr\{X_n = x \mid \tau^\rho(n) = \tau\}$ is 3-times differentiable on I_0 as a function of τ, and therefore also the expression (17). This entails the Lipschitz property of $S(\tau)$ on the compact set I_0 and finishes the verification of (iii). Finally, since, for a component a of the vector τ,

$$| - \tau_a^\rho(n) + I(a \in X_n) f(X_n)| \leq |\tau_a^\rho(n)| + |f(X_n)| \leq M_1 + M_2,$$

where $M_1 = \sup_{\tau \in I_0} ||\tau|| < \infty$ and $M_2 = \max\{f(x) \mid x \in \{0,1\}^m\} < \infty$, we have

$$||H^\rho(n)||^3 = \left(\sum_a (H_a^\rho(n))^2\right)^{3/2} \leq (2m)^{3/2} \cdot (M_1 + M_2)^3,$$

which shows condition (iv). □

Proof of Lemma 5.1. Induction with respect to n. By assumption, the inequalities hold for $n = 0$. Let $n \geq 0$. Then, by adding the two equations

$$\tau_i^{(k)}(n+1) = (1-\rho)\tau_i^{(k)}(n) + \rho\, I(X_n \text{ makes move } k \text{ on position } i)\, f(X_n) \quad (k = 0, 1),$$

we obtain

$$\tau_i^{(0)}(n+1) + \tau_i^{(1)}(n+1) = (1-\rho)\,(\tau_i^{(0)}(n) + \tau_i^{(1)}(n)) + \rho\, f(X_n) \geq (1-\rho) \cdot c + \rho \cdot c = c.$$

Analogously, the second inequality is verified. □

Proof of Proposition 5.3. Without loss of generality, we consider OneMax. It is easy to see from the symmetry properties of OneMax and from (8) that equal initial pheromone values at all positions i,

$$\tau_1^{(k)}(0) = \ldots = \tau_m^{(k)}(0) \quad (k = 0, 1)$$

entail that in the ACDP,

$$\tau_1^{(k)}(t) = \ldots = \tau_m^{(k)}(t) \quad (k = 0, 1) \tag{18}$$

holds also for each time $t \geq 0$. As a consequence, also the expected passage fitness values for the up-moves are equal in each position, and so are the expected passage fitness values for the down-moves. For abbreviation, we set $u = \tau_1^{(1)}(t)$, $w = \tau_1^{(0)}(t)$, $F_u = F_{\underline{0},1}(\tau(t))$, and $F_w = F_{\underline{0},-1}(\tau(t))$. Note that all these quantities depend on t. From (14), (15) and (18), we get

$$F_u = \frac{u((m+c)u + (1+c)w)}{(u+w)^2}, \quad F_w = \frac{w((m-1+c)u + cw)}{(u+w)^2}. \tag{19}$$

Thus, by (8),

$$\dot{u} = F_u - u = u\left[\frac{(m+c)u + (1+c)w}{(u+w)^2} - 1\right], \quad \dot{w} = F_w - w = w\left[\frac{(m-1+c)u + cw}{(u+w)^2} - 1\right]. \tag{20}$$

It has been shown in [17] that if $\Phi = \Phi(t)$ denotes the total pheromone flow out of the start node $\underline{0}$ at time t, i.e., the sum of all pheromone values on arcs leaving $\underline{0}$, then the expected fitness μ of an ant's walk based on the pheromone vector at time t in the ACDP is given by $\mu = \Phi + \dot{\Phi}$. Since $\Phi = u + w$ for the chain, we get by short calculation

$$\mu = F_u + F_w = \frac{mu}{u+w} + c.$$

We transform u, w and μ to *relative* values by setting

$$\xi = u/m, \quad \eta = w/m, \quad \varphi = \mu/m,$$

and we abbreviate $1/m$ by δ, such that

$$\dot{\xi} = \xi\left[\frac{(1+c\delta)\xi + (1+c)\delta\eta}{(\xi+\eta)^2} - 1\right], \quad \dot{\eta} = \eta\left[\frac{(1+(c-1)\delta)\xi + c\delta\eta}{(\xi+\eta)^2} - 1\right], \quad \varphi = \frac{\xi}{\xi+\eta} + c\delta, \tag{21}$$

and hence

$$\dot{\varphi} = \frac{\dot{\xi}\eta - \xi\dot{\eta}}{(\xi+\eta)^2} = \frac{\xi\eta}{(\xi+\eta)^4}\left\{(1+c\delta)\xi + (c+1)\delta\eta - (1+(c-1)\delta)\xi - c\delta\eta\right\} = \frac{\delta\xi\eta}{(\xi+\eta)^3}. \tag{22}$$

Alternatively, (22) can be written as

$$\dot{\varphi} = \frac{\delta}{\xi+\eta} \cdot \frac{\xi}{\xi+\eta} \cdot \frac{\eta}{\xi+\eta} = \frac{\delta}{\xi+\eta}\,(\varphi - c\delta)\,(1 - (\varphi - c\delta)).$$

Let us set $\varphi_c = \varphi - c/m = \varphi - c\delta$. Then the last equation simplifies to

$$\dot{\varphi}_c = \frac{\delta}{\xi+\eta}\,\varphi_c\,(1 - \varphi_c). \tag{23}$$

with $\varphi_c(0) = 1/2$. Since $f_{max} = m + c$ and $f_{min} = c$, the expected relative fitness is just $(\mu - c)/m = \varphi - c\delta = \varphi_c$.

Initially, we have

$$\xi(0) = \eta(0) = (1 + c\delta)/2 > 0.$$

By (21), $\dot{\xi}(t) = 0$ if $\xi(t) = 0$, so $\xi(t) \geq 0$ for all t, and similarly $\eta(t) \geq 0$ for all t.

The solution of eq. (23) is

$$\varphi_c(t) = \left[1 + \exp\left(-\int_0^t \frac{d\sigma}{u(\sigma) + w(\sigma)}\right)\right]^{-1}.$$

If $\mu_l \leq u(t) + w(t) \leq \mu_r$ for all t, it follows:

$$\left(1 + \exp\left(-\frac{t}{\mu_r}\right)\right)^{-1} \leq \varphi_c(t) \leq \left(1 + \exp\left(-\frac{t}{\mu_l}\right)\right)^{-1}.$$

The times when the first, second and third function, respectively, in the above inequality are equal to $1 - \epsilon$, satisfy

$$\mu_l \log((1-\epsilon)/\epsilon) \leq t_\varphi \leq \mu_r \log((1-\epsilon)/\epsilon).$$

We will now derive expressions for μ_l and μ_r.

Let $z = u + w$. Straightforward calculation shows

$$\dot{z} = \dot{u} + \dot{w} = \frac{m}{1 + w/u} + c - z. \tag{24}$$

Furthermore, one easily verifies

$$\frac{d}{dt}\frac{w}{u} = \frac{\dot{w}u - w\dot{u}}{u^2} = \frac{1}{u^2}\left[(F_w - w)u - w(F_u - u)\right] = -\frac{w}{u(u+w)} < 0,$$

so $w(t)/u(t) \leq w(0)/u(0) = 1$ for all t.

The initial condition for eq. (24) is $z(0) = m + c$. Since $0 \leq w/u \leq 1$, it follows $m/2 + c \leq m/(1 + w/u) + c \leq m + c$. The function $\tilde{z} = (m + c)(1 + e^{-t})$ is solution of

$$\frac{d\tilde{z}}{dt} = m + c - \tilde{z}, \tag{25}$$

with initial condition $\tilde{z}(0) = 2(m+c)$. Subtracting from each of the two sides of (25) the corresponding sides of eq. (24), it follows

$$\frac{d}{dt}(\tilde{z} - z) = m + c - (m/(1 + w/u) + c) - (\tilde{z} - z) \geq -(\tilde{z} - z).$$

If we introduce $r = \tilde{z} - z$, the inequality above gives $\dot{r} \geq -r$, with initial condition $r(0) = m + c > 0$. It is known that this implies $r(t) \geq 0$. Therefore, $\tilde{z} \geq z$, and since $\tilde{z} \leq 2(m+c)$, we have $z \leq 2(m+c)$, which provides an expression for μ_r.

Analogously, we can consider the function $z' = m/2 + c(1 - e^{-t})$, which satisfies the equation

$$\frac{dz'}{dt} = m/2 + c - z', \tag{26}$$

with initial condition $z'(0) = m/2$. Subtracting each of the two sides of (26) from the corresponding sides of eq. (24), it follows

$$\frac{d}{dt}(z - z') = (m/(1 + w/u) + c) - (m/2 + c) - (z - z') \geq -(z - z').$$

This implies in this case $z \geq z'$, and since $z' \geq m/2$, we have $m/2 \leq z$, which provides an expression for μ_l. □