UNIVERSITY OF DORTMUND

REIHE COMPUTATIONAL INTELLIGENCE

COLLABORATIVE RESEARCH CENTER 531

Design and Management of Complex Technical Processes and Systems by means of Computational Intelligence Methods

Comparing Variants of MMAS ACO Algorithms on Pseudo-Boolean Functions

Frank Neumann, Dirk Sudholt and Carsten Witt

No. CI-230/07

Technical Report ISSN 1433-3325 May 2007

Secretary of the SFB 531 · University of Dortmund · Dept. of Computer Science/LS 2
44221 Dortmund · Germany

This work is a product of the Collaborative Research Center 531, "Computational Intelligence," at the University of Dortmund and was printed with financial support of the Deutsche Forschungsgemeinschaft.

Comparing Variants of MMAS ACO Algorithms on Pseudo-Boolean Functions

Frank Neumann
Max-Planck-Institut für Informatik
66123 Saarbrücken, Germany

Dirk Sudholt* Carsten Witt*
FB Informatik, LS 2
Universität Dortmund
44221 Dortmund, Germany

May 30, 2007

Abstract

Recently, the first rigorous runtime analyses of ACO algorithms have been presented. These results concentrate on variants of the MAX-MIN ant system by Stützle and Hoos and consider their runtime on simple pseudo-Boolean functions such as OneMax and LeadingOnes. Interestingly, it turns out that a variant called 1-ANT is very sensitive to the choice of the evaporation factor while a recent technical report by Gutjahr and Sebastiani suggests partly opposite results for their variant called MMAS. In this paper, we elaborate on the differences between the two ACO algorithms, generalize the techniques by Gutjahr and Sebastiani and show improved results.

1 Introduction

Randomized search heuristics have been shown to be good problem solvers with various application domains. Two prominent examples belonging to this class of algorithms are Evolutionary Algorithms (EAs) and Ant Colony Optimization (ACO) (Dorigo and Stützle, 2004). Especially ACO algorithms have been shown to be very successful for solving problems from combinatorial optimization. Indeed, the first problem where an ACO algorithm has been applied was the Traveling Salesperson Problem (TSP) (Dorigo, Maniezzo and Colorni, 1991) which is one of the most studied combinatorial problems in computer science.

In contrast to many successful applications, theory lags far behind the practical evidence of all randomized search heuristics. In particular in the case of ACO algorithms, the analysis of such algorithms with respect to their runtime behavior has been started only recently. The analysis of randomized search heuristics

*This author was supported by the Deutsche Forschungsgemeinschaft (SFB) as a part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

(e. g., Droste, Jansen and Wegener, 2002) is carried out as in the classical algorithm community and makes use of several strong methods for the analysis of randomized algorithms (Motwani and Raghavan, 1995; Mitzenmacher and Upfal, 2005).

Regarding ACO, only convergence results (Gutjahr, 2002) were known until 2006 and analyzing the runtime of ACO algorithms has been pointed out as a challenging task by Dorigo and Blum (2005). First steps into analyzing the runtime of ACO algorithms have been made by Gutjahr (2007a), and, independently, the first theorems on the runtime of a simple ACO algorithm called 1-ANT have been obtained at the same time by Neumann and Witt (2006). Later on this algorithm has been further investigated for the optimization of some well-known pseudo-Boolean functions (Doerr, Neumann, Sudholt, and Witt, 2007). A conclusion from these investigations is that the 1-ANT is very sensitive w. r. t. the choice of the evaporation factor ρ. Increasing the value of ρ only by a small amount may lead to a phase transition and turn an exponential runtime into a polynomial one. In contrast to this, a simple ACO algorithm called MMAS_{bs} has been investigated in a recent report by Gutjahr and Sebastiani (2007) where this phase transition does not occur. Gutjahr (2007b) conjectures that the different behavior of MMAS_{bs} and 1-ANT is due to their slightly different replacement strategies: MMAS_{bs} accepts only strict improvements while 1-ANT accepts also equal-valued solutions. We will however show that the replacement strategies do not explain the existence of the phase transition. Instead, the reason is that the 1-ANT only updates pheromone values when the best-so-far solution is replaced.

This motivates us to study MMAS variants where the pheromone values are updated in each iteration. First, we consider the MMAS algorithm by Gutjahr and Sebastiani (2007) and show improved and extended results. In particular, we make use of the method called fitness-based partitions which is well-known from the analysis of evolutionary algorithms. Additionally, we study plateau functions and argue why the replacement strategy of the 1-ANT combined with persistent pheromone updates is more natural. Investigating the function Needle, we show that this can reduce the runtime of the ACO algorithm significantly.

The outline of the paper is as follows. In Section 2, we introduce the algorithms that are subject of our investigations. The behavior of these algorithms on well-known plateau functions is considered in Section 3, and Section 4 deals with analyses for some popular unimodal functions. We finish with some conclusions.

2 Algorithms

We consider the runtime behavior of two ACO algorithms. Solutions for a given problem, in this paper bit strings $x \in \{0,1\}^n$ for pseudo-boolean functions $f\colon \{0,1\}^n \to \mathbb{R}$, are constructed by a random walk on a so-called construction

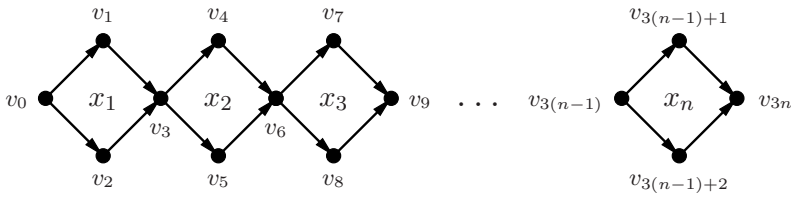

Figure 1: Construction graph for pseudo-Boolean optimization

graph $C = (V, E)$, which is a directed graph with a designated start vertex $s \in V$ and pheromone values $\tau\colon E \to \mathbb{R}$ on the edges.

Algorithm 1 (Construct(C, τ))

1.) v:=s, mark v as visited.

2.) Let N_v be the set of non-visited successors of v in C. If $N_v \neq \emptyset$:

a.) Choose successor $w \in N_v$ with probability $\tau_{(v,w)} / \sum_{(v,u)|u\in N_v} \tau_{(v,u)}$.

b.) Mark w as visited, set $v := w$ and go to 2.).

3.) Return the solution x and the path $P(x)$ constructed by this procedure.

We examine the construction graph given in Figure 1, which is known in the literature as *Chain* (Gutjahr, 2006). For bit strings of length n, the graph has $3n + 1$ vertices and $4n$ edges. The decision whether a bit x_i, $1 \leq i \leq n$, is set to 1 is made at node $v_{3(i-1)}$. If edge $(v_{3(i-1)}, v_{3(i-1)+1})$ (called 1-edge) is chosen, x_i is set to 1 in the constructed solution. Otherwise the corresponding 0-edge is taken, and $x_i = 0$ holds. After this decision has been made, there is only one single edge which can be traversed in the next step. In case that $(v_{3(i-1)}, v_{3(i-1)+1})$ has been chosen, the next edge is $(v_{3(i-1)+1}, v_{3i})$, and otherwise the edge $(v_{3(i-1)+2}, v_{3i})$ will be traversed. Hence, these edges have no influence on the constructed solution and we can assume $\tau_{(v_{3(i-1)},v_{3(i-1)+1})} = \tau_{(v_{3(i-1)+1},v_{3i})}$ and $\tau_{(v_{3(i-1)},v_{3(i-1)+2})} = \tau_{(v_{3(i-1)+2},v_{3i})}$ for $1 \leq i \leq n$. We ensure that $\sum_{(u,\cdot)\in E} \tau_{(u,\cdot)} = 1$ for $u = v_{3i}$, $0 \leq i \leq n-1$, and $\sum_{(\cdot,v)} \tau_{(\cdot,v)} = 1$ for $v = v_{3i}$, $1 \leq i \leq n$. Let $p_i = \mathrm{Prob}(x_i = 1)$ be the probability of setting the bit x_i to 1 in the next constructed solution. Due to our setting $p_i = \tau_{(3(i-1),3(i-1)+1)}$ and $1 - p_i = \tau_{(3(i-1),3(i-1)+2)}$ holds, i. e., the pheromone values correspond directly to the probabilities for choosing the bits in the constructed solution. In addition, following the MAX-MIN ant system by Stützle and Hoos (2000), we restrict each $\tau_{(u,v)}$ to the interval $[1/n, 1 - 1/n]$ such that every solution always has a positive probability of being chosen.

Depending on whether edge (u, v) is contained in the path $P(x)$ of the constructed solution x, the pheromone values are updated to τ' in the update procedure as follows:

$$\tau'_{(u,v)} = \begin{cases} \min\{(1-\rho)\cdot\tau_{(u,v)} + \rho,\ \ 1-1/n\} & \text{if } (u,v) \in P(x) \\ \max\{(1-\rho)\cdot\tau_{(u,v)},\ \ 1/n\} & \text{otherwise.} \end{cases}$$

The following algorithm, which we call MMAS*, has been defined by Gutjahr and Sebastiani (2007) under the original name MMAS_{bs}. Here, in each iteration the best solution obtained during the run of the algorithm, called best-so-far solution, is rewarded. Another property of the model is that the best-so-far solution may not switch to another one of the same fitness.

Algorithm 2 (MMAS*)

1.) Set $\tau_{(u,v)} = 1/2$ *for all* $(u, v) \in A$.

2.) Compute a solution x *using Construct*(C, τ).

3.) Update the pheromone values and set $x^* := x$.

4.) Compute a solution x *using Construct*(C, τ).

5.) If $f(x) > f(x^*)$, *set* $x^* := x$

6.) Update the pheromone values with respect to x^*.

7.) Go to 4.).

Using this model, it is much easier to adapt results from the well-known evolutionary algorithm called (1+1) EA than in the case of the 1-ANT, i. e., the MAX-MIN ACO variant examined by Neumann and Witt (2006) and Doerr, Neumann, Sudholt, and Witt (2007). In particular, many results using the technique of fitness-based partitions may be transferred to MMAS* by taking into account an additional amount of time until the best solution has been rewarded such that an improvement is obtained with a large enough probability (see below for details). This is only possible since pheromone updates occur in each iteration. In contrast to this, the 1-ANT only updates the pheromone values if the best-so-far solution is replaced by solutions of at least the same fitness, i. e., steps 5.) and 6.) in the above description are substituted by

> *If* $f(x) \geq f(x^*)$, *set* $x^* := x$ *and update the pheromone values w. r. t.* x^*.

This update strategy may lead to a large discrepancy between the expected value of the next solution and the currently best one and is the main reason for exponential runtimes of the 1-ANT even for really simple functions and relatively large values of ρ (see Neumann and Witt, 2006; Doerr, Neumann, Sudholt, and Witt, 2007).

If the value of ρ is chosen large enough in MMAS*, the pheromone borders $1/n$ or $1 - 1/n$ are touched for every bit of the rewarded solution. In this case, MMAS* equals the algorithm called (1+1) EA*, which is known from the analysis of evolutionary algorithms (Jansen and Wegener, 2001).

Algorithm 3 ((1+1) EA*)

1. *Choose an initial solution* $x^* \in \{0,1\}^n$ *uniformly at random.*
2. *Repeat*
   a) *Create* x *by flipping each bit of* x^* *with probability* $1/n$.
   b) *If* $f(x) > f(x^*)$, *set* $x^* := x$.

As already pointed out in (Jansen and Wegener, 2001), the (1+1) EA* has difficulties with simple plateaus of constant fitness as no search points of the same fitness as the so far best one are accepted. Accepting solutions with equal fitness enables the algorithm to explore plateaus by random walks. Therefore, it seems more natural to replace search points by new solutions that are at least as good. In the case of evolutionary algorithms, this leads to the well-known (1+1) EA which differs from the (1+1) EA* only in step 2.b) of the algorithm. Similarly, we derive MMAS from MMAS* using this acceptance condition. It should be noted that MMAS is *not* just a variant of the 1-ANT with different pheromone values since it still updates pheromones in each iteration.

Algorithm 4 (Acceptance condition for the (1+1) EA and MMAS)
– *If* $f(x) \geq f(x^*)$, *set* $x^* := x$.

In the remainder of the paper, we will examine the behavior of MMAS compared to MMAS*. For the analysis of an algorithm, we consider the number of solutions that are constructed by the algorithm until an optimal solution has been obtained for the first time. This is called the *optimization time* of the algorithm and is a well-accepted measure in the analysis of evolutionary algorithms since each point of time corresponds to a fitness evaluation. Often the expectation of this value is considered and called the *expected optimization time*.

Before we derive the first results, it is helpful to introduce the quantity that informally has been mentioned above. Suppose there is a phase such that MMAS or MMAS* never replaces the best-so-far solution x^* in step 5.) of the algorithm. This implies that the best-so-far solution is rewarded again and again until all pheromone values have reached their upper or lower borders corresponding to the setting of the bits in x^*. The advantage is that probabilities of improvements can be estimated more easily as soon as x^* has been "frozen in pheromone" this way. Gutjahr and Sebastiani (2007) call the time for this to happen t^* and bound it from above by $-\ln(n-1)/\ln(1-\rho)$. This holds since a pheromone value which is only increased during t steps is at least $\min\{1-1/n, 1-(1-1/n)(1-\rho)^t\}$ after the iterations, pessimistically assuming the worst-case initialization $1/n$ for this value and $1-1/n$ for the complementary pheromone value. In the present paper, we use $\ln(1-\rho) \leq -\rho$ for $0 \leq \rho \leq 1$ and arrive at the handy upper bound

$$t^* \leq \frac{\ln n}{\rho}. \tag{1}$$

3 Plateau Functions

Plateaus are regions in the search space where all search points have the same fitness. Consider a function $f\colon \{0,1\}^n \to \mathbb{R}$ and assume that the number of different objective values for that function is D. Then there are at least $2^n/D$ search points with the same objective vector. Often, the number of different objective values for a given function is polynomially bounded. This implies an exponential number of solutions with the same objective value. In the extreme case, we end up with the function NEEDLE where only one single solution has objective value 1 and the remaining ones get value 0. The function is defined as

$$\textsc{Needle}(x) := \begin{cases} 1 & \text{if } x = x_{\text{OPT}}, \\ 0 & \text{otherwise,} \end{cases}$$

where x_{OPT} is the unique global optimum. Gutjahr and Sebastiani (2007) compare MMAS* and (1+1) EA* w. r. t. their runtime behavior. For suitable values of ρ that are exponentially small in n, the MMAS* has expected optimization time $O(c^n)$, $c \geq 2$ an appropriate constant, and beats the (1+1) EA*. The reason is that MMAS* behaves nearly as random search on the search space while the initial solution of the (1+1) EA* has Hamming distance n to the optimal one with probability 2^{-n}. To obtain from such a solution an optimal one, all n bits have to flip, which has expected waiting time n^n, leading in summary to an expected optimization time $\Omega((n/2)^n)$. In the following, we show a similar result for MMAS* if ρ decreases only polynomially with the problem dimension n.

Theorem 1 *Choosing $\rho = 1/\text{poly}(n)$, the optimization time of MMAS* on* NEEDLE *is at least $(n/6)^n$ with probability $1 - e^{-\Omega(n)}$.*

Proof: Let x be the first solution constructed by MMAS* and denote by x_{OPT} the optimal one. As it is chosen uniformly at random from the search space, the expected number of positions where x and x_{OPT} differ is $n/2$ and there are at least $n/3$ such positions with probability $1 - e^{-\Omega(n)}$ using Chernoff bounds. At these positions the values of x are rewarded as long as the optimal solution has not been obtained. This implies that the probability to obtain the optimal solution in the next step is at most $2^{-n/3}$. After at most $t^* \leq (\ln n)/\rho$ (see Inequality (1)) iterations, the pheromone values of x have touched their borders provided x_{OPT} has not been obtained. The probability of having obtained x_{OPT} within a phase of t^* steps is at most $t^* \cdot 2^{-n/3} = e^{-\Omega(n)}$. Hence, the probability to produce a solution that touches its pheromone borders and differs from x_{OPT} in at least $n/3$ positions before producing x_{OPT} is $1 - e^{-\Omega(n)}$. In this case, the expected number of steps to produce x_{OPT} is $(n/3)^n$ and the probability of having reached this goal within $(n/6)^n$ steps is at most 2^{-n}. □

The probability to choose an initial solution x that differs from x_{OPT} by n positions is 2^{-n}, and in this case, after all n bits have reached their corresponding pheromone borders, the probability to create x_{OPT} equals n^{-n}. Using the

ideas of Theorem 1 the following corollary can be proved which asymptotically matches the lower bound for the (1+1) EA* given in Gutjahr and Sebastiani (2007).

Corollary 1 *Choosing* $\rho = 1/\text{poly}(n)$*, the expected optimization time of MMAS* on* NEEDLE *is* $\Omega((n/2)^n)$.

It is well known that the (1+1) EA that accepts each new solution has expected optimization time $2^{n+o(n)}$ on NEEDLE (see Garnier, Kallel and Schoenauer, 1999; Wegener and Witt, 2005) even though it samples with high probability in the Hamming neighborhood of the latest solution. On the other hand, MMAS* will have a much larger optimization time unless ρ is superpolynomially small (Theorem 1). In the following, we will show that MMAS is competitive with the (1+1) EA even for large ρ-values.

Theorem 2 *Choosing* $\rho = \Omega(1)$*, the expected optimization time of MMAS on* NEEDLE *is* $2^{n+o(n)}$.

Proof: By the symmetry of the construction procedure and uniform initialization, we w. l. o. g. assume that the needle x_{OPT} equals the all-ones string 1^n. As in Wegener and Witt (2005), we study the process on the constant function $f(x) = 0$. The first hitting times for the needle are the same on NEEDLE and the constant function while the invariant limit distribution for the constant function is easier to study since it is uniform over the search space.

The proof idea is to study a kind of "mixing time" $t(n)$ after which each bit is independently set to 1 with a probability of at least $1/2 - 1/n$ regardless of the initial pheromone value on its 1-edge. This implies that the probability of creating the needle is at least $(1/2 - 1/n)^n \geq e^{-3}2^{-n}$ (for n large enough) in some step after at most $t(n)$ iterations. We successively consider independent phases of length $t(n)$ until the needle is sampled. Estimating by a geometrically distributed waiting time, the expected optimization time is bounded by $O(t(n) \cdot 2^n)$. The theorem follows if we can show that $t(n) = \text{poly}(n)$.

To bound $t(n)$, we note that the limit distribution of each pheromone value is symmetric with expectation $1/2$, hence, each bit is set to 1 with probability $1/2$ in the limit. We consider a coupling of two independent copies of the Markov chain for the pheromone values of a bit such that we start the chain from two different states. The task is to bound the coupling time $c(n)$, defined as the first point of time where both chains meet in the same border state $1/n$ or $1 - 1/n$ (taking a supremum over any two different initial states). If we know that $E(c(n))$ is finite, then Markov's inequality yields that $c(n) \leq nE(c(n))$ with probability at least $1 - 1/n$, and the coupling lemma (Mitzenmacher and Upfal, 2005) implies that the total variation distance to the limit distribution is at most $1/n$ at time $nE(c(n))$. Hence, the bit is set to 1 with probability at least $1/2 - 1/n$ then and we can use $t(n) := nE(c(n))$.

It remains to prove $E(c(n)) = \text{poly}(n)$ for the considered coupling. Since $\rho = \Omega(1)$, the pheromone value of a bit reaches one of its borders in $t^* = O(\log n)$

(Inequality (1)) iterations with probability at least $\prod_{t=0}^{t^*}(1-(1-1/n)(1-\rho)^t) = \Omega(1/n)$ (using the same estimations for pheromone values as Gutjahr and Sebastiani, 2007). Hence, with probability 1/2, one chain reaches a border in $O(n \log n)$ iterations. A pheromone value that is at a border does not change within $O(n \log n)$ iterations with probability at least $(1-1/n)^{O(n \log n)} = 1/\text{poly}(n)$. In this phase, the pheromone value of the other chain reaches this border with probability $\Omega(1)$. Repeating independent phases, we have bounded $E(c(n))$ by a polynomial. □

The function NEEDLE requires an exponential optimization time for each algorithm that has been considered. Often plateaus are much smaller, and randomized search heuristics have a good chance to leave them within a polynomial number of steps. Gutjahr and Sebastiani (2007) consider the function NH-ONEMAX that consists of the NEEDLE-function on $k = \log n$ bits and the function ONEMAX on $n-k$ bits, which can only be optimized if the needle has been found on the needle part. The function is defined as

$$\text{NH-ONEMAX}(x) = \left(\prod_{i=1}^{k} x_i\right)\left(\sum_{i=k+1}^{n} x_i\right).$$

Using the ideas in the proof of Theorem 1 and taking into account the logarithmic size of the needle of NH-ONEMAX, MMAS* with polylogarithmically small ρ finds the needle only after an expected superpolynomial number $2^{\Omega(\log^2 n)}$ of steps, and the following theorem can be shown.

Theorem 3 *Choosing $\rho = 1/\text{polylog}(n)$, the expected optimization time of MMAS* on* NH-ONEMAX *is $2^{\Omega(\log^2 n)}$.*

As the needle part only consists of $\log n$ bits, MMAS can find the needle after an expected polynomial number of steps (Theorem 2). After this goal has been achieved, the unimodal function ONEMAX has to be optimized. Together with our investigations for unimodal functions carried out in the next section (in particular the upper bound from Theorem 6), the following result can be proved.

Theorem 4 *Choosing $\rho = \Omega(1)$, the expected optimization time of MMAS on* NH-ONEMAX *is polynomial.*

4 Unimodal Functions, OneMax and LeadingOnes

Gutjahr and Sebastiani (2007) extend the well-known fitness-level method, also called the method of f-based partitions, from the analysis of evolutionary algorithms (see, e. g., Wegener, 2002) to their considered MMAS-algorithm, i. e., the MMAS*. Let $A_1, \ldots, A_m$ be an f-based partition w. r. t. the pseudo-Boolean

fitness function $f\colon \{0,1\}^n \to \mathbb{R}$, i. e., for any pair of search points $x \in A_i, y \in A_j$ where $j > i$ it holds $f(x) < f(y)$, and A_m contains only optimal search points. Moreover, let s_i, $1 \leq i \leq m-1$, be a lower bound on the probability of the (1+1) EA (or, in this case equivalently, the (1+1) EA*) leaving set A_i. Using the quantity t^*, the expected runtime of MMAS* on f is bounded from above by

$$\sum_{i=1}^{m-1} \left(t^* + \frac{1}{s(i)} \right)$$

(which is a special case of Eq. (13) by Gutjahr and Sebastiani, 2007). Since $t^* \leq (\ln n)/\rho$, we obtain the more concrete bound

$$\frac{m \ln n}{\rho} + \sum_{i=1}^{m-1} \frac{1}{s(i)}, \tag{2}$$

in which the right-hand sum is exactly the upper bound obtained w. r. t. the (1+1) EA and (1+1) EA*. To prove the bound (2) for MMAS*, it is essential that equally good solutions are rejected (see Gutjahr and Sebastiani, 2007, for a formal derivation). Informally speaking: for each fitness-level, MMAS* in the worst case has to wait until all pheromone values have their obtained upper and lower borders such that the best-so-far solution is "frozen in pheromone" and the situation is like in the (1+1) EA* with the best-so-far solution as the current search point.

In the technical report by Gutjahr and Sebastiani (2007), the proposed fitness-level method is basically applied in the context of the unimodal functions OneMax and LeadingOnes. Our aim is to show how the method can be applied to arbitrary unimodal functions both w. r. t. MMAS and MMAS*. Moreover, we generalize the upper bounds obtained by Gutjahr and Sebastiani (2007) for the example functions OneMax and LeadingOnes and, for the first time, we show a lower bound on the expected optimization time of MMAS* on LeadingOnes. This allows us to conclude that the fitness-level method can provide almost tight upper bounds.

4.1 General Results

Unimodal functions are a well-studied class of fitness functions in the literature on evolutionary computation (e. g., Droste, Jansen and Wegener, 2002). For the sake of completeness, we repeat the definition of unimodality for pseudo-Boolean fitness functions.

Definition 1 *A function $f\colon \{0,1\}^n \to \mathbb{R}$ is called unimodal if there exists for each non-optimal search point x a Hamming neighbor x' where $f(x') > f(x)$.*

Unimodal functions are often believed to be easy to optimize. This holds if the set of different fitness values is not too large. In the following, we consider

unimodal functions attaining D different fitness values. Such a function is optimized by the (1+1) EA and (1+1) EA* in expected time $O(nD)$. This bound is transferred to MMAS* by the following theorem.

Theorem 5 *The expected optimization time of MMAS* on a unimodal function attaining D different fitness values is $O((n + \log n/\rho)D)$.*

Proof: By the introductory argument, we only have to set up an appropriate fitness-based partition. We choose the D sets of preimages of different fitness values. By the unimodality, there is for each current search point x a better Hamming neighbor x' of x in a higher fitness-level set. The probability of the (1+1) EA (or, equivalently, MMAS* with all pheromone values at a border) to produce x' in the next step is $\Omega(1/n)$. By (2), this completes the proof. □

MMAS differs from MMAS* by accepting solutions that are at least as good as the best solution obtained during the optimization process. Hence, the best-so-far solution may switch among several solutions with the same fitness value, and, on every fitness-level, pheromone values may perform random walks between the upper and lower pheromone borders. In comparison to MMAS*, this behavior makes it harder to bound the expected time for an improvement, and the following upper bound is worse than the upper bound for MMAS*.

Theorem 6 *The expected optimization time of MMAS on a unimodal function attaining D different fitness values is $O((n^2(\log n)/\rho)D)$.*

Proof: We only need to show that the expected time for an improvement is $O(n^2(\log n)/\rho)$. The probability that MMAS produces within $O((\log n)/\rho)$ steps a solution being at least as good as (not necessarily better than) the best-so-far solution x^* is $\Omega(1)$ since after at most $(\ln n)/\rho$ steps without exchanging x^* all pheromone values have touched their bounds and then the probability of rediscovering x^* is $\Omega(1)$. We now show that the conditional probability of an improvement if x^* is replaced is $\Omega(1/n^2)$.

Let $x_1, \ldots, x_m$ be an enumeration of all solutions with fitness values equal to the best-so-far fitness value. Due to unimodality, each x_i, $1 \leq i \leq m$, has some better Hamming neighbor y_i; however, the y_i need not be disjoint. Let X and Y denote the event to generate some x_i or some y_i, resp., in the next step. In the worst case $y_1, \ldots, y_m$ are the only possible improvements, hence the theorem follows if we can show $\text{Prob}(Y \mid X \cup Y) \geq 1/n^2$ which is equivalent to $\text{Prob}(Y) \geq \text{Prob}(X)/(n^2 - 1)$.

If $p(x_i)$ is the probability to construct x_i, we have $p(x_i)/p(y_i) \leq (1-\frac{1}{n})/\frac{1}{n} = n - 1$ as the constructions only differ in one bit. Each y_i may appear up to n times in the sequence $y_1, \ldots, y_m$, hence $\text{Prob}(Y) \geq \frac{1}{n}\sum_{i=1}^{m} \text{Prob}(y_i)$ and

$$\text{Prob}(X) = \sum_{i=1}^{m} p(x_i) \leq (n-1) \cdot \sum_{i=1}^{m} p(y_i) \leq n(n-1) \cdot \text{Prob}(Y).$$

Therefore, $\text{Prob}(Y) \geq \text{Prob}(X)/(n^2 - 1)$ follows. □

Theorems 5 and 6 show that the expected optimization times of both MMAS and MMAS* are polynomial for all unimodal functions as long as $D = \text{poly}(n)$ and $\rho = 1/\text{poly}(n)$. Since MMAS and 1-ANT do not differ in their replacement strategy, this disproves the conjecture by Gutjahr (2007b) that accepting equally good solutions leads to the phase transition behavior from polynomial to exponential runtimes of the 1-ANT on the following example functions.

4.2 OneMax

The probably most often studied example function in the literature on evolutionary computation is the unimodal function $\textsc{OneMax}(x) = x_1 + \cdots + x_n$. In the runtime analysis of the 1-ANT on OneMax by Neumann and Witt (2006), it is shown that there exists a threshold value for ρ (in our notation basically $\rho = O(1/n^\epsilon)$ for some small constant $\epsilon > 0$) below which no polynomial runtime is possible. As argued above, due to Theorems 5 and 6, this phase transition can occur neither with MMAS* nor with MMAS. We are interested in improved upper bounds for the special case of MMAS* on OneMax. The following theorem has already been proven for some values of ρ by Gutjahr and Sebastiani (2007). We however obtain polynomial upper bounds for all ρ bounded by some inverse polynomial.

Theorem 7 *The expected optimization time of MMAS* on* OneMax *is bounded from above by* $O((n \log n)/\rho)$.

Proof: The proof is an application of the above-described fitness-level method with respect to the partition $A_i = \{x \mid f(x) = i\}$, $0 \leq i \leq n$. Using the arguments by Gutjahr and Sebastiani (2007)—or, equivalently, the upper bound on the expected runtime of the (1+1) EA on OneMax (Droste, Jansen and Wegener, 2002)—we obtain that the term $\sum_{i=0}^{m-1} 1/(s(i))$ is $O(n \log n)$. Using (2), the upper bound $O((n \log n)/\rho)$ follows. □

The bound is never better than $\Theta(n \log n)$, which is the expected runtime of the (1+1) EA and (1+1) EA* on OneMax. At the moment, we are not able to show a matching lower bound $\Omega(n \log n)$ on the expected optimization time of MMAS*; however, we can show that the expected optimization time is growing with respect to $1/\rho$ as the upper bound suggests. We state our result in a more general framework: as known from the considerations by Droste, Jansen and Wegener (2002), the mutation probability $1/n$ of the (1+1) EA is optimal for many functions including OneMax. One argument is that the probability mass has to be quite concentrated around the best-so-far solution to allow the (1+1) EA to rediscover the last accepted solution with good probability. Given a mutation probability of $\alpha(n)$, this probability of rediscovery equals $(1-\alpha(n))^n$, which converges to zero unless $\alpha(n) = O(1/n)$. The following lemma exploits the last observation for a general lower bound on the expected optimization time of both MMAS and MMAS*.

Theorem 8 *Let $f\colon \{0,1\}^n \to \mathbb{R}$ be a function with a unique global optimum. Choosing $\rho = 1/\mathrm{poly}(n)$, the expected optimization time of MMAS and MMAS* on f is $\Omega((\log n)/\rho)$.*

Proof: W. l. o. g., 1^n is the unique optimum. If, for each bit, the success probability (defined as the probability of creating a one) is bounded from above by $1 - 1/\sqrt{n}$ then the solution 1^n is created with only exponentially small probability $(1 - 1/\sqrt{n})^n \leq e^{-\sqrt{n}}$. Using the uniform initialization and pheromone update formula of MMAS and MMAS*, the success probability of a bit after t steps is bounded from above by $1 - (1-\rho)^t/2$. Hence, all success probabilities are bounded as desired within $t = (\ln(n/4))/(2\rho)$ steps since

$$1 - \frac{1}{2}(1-\rho)^t \;\leq\; 1 - \frac{e^{-(\ln n - \ln 4)/2}}{2} \;=\; 1 - \frac{1}{\sqrt{n}}.$$

Since $\rho = 1/\mathrm{poly}(n)$ and, therefore $t = \mathrm{poly}(n)$, the total probability of creating the optimum in t steps is still at most $te^{-\sqrt{n}} = e^{-\Omega(\sqrt{n})}$, implying the lower bound on the expected optimization time. □

Hence, the expected optimization time of MMAS* on OneMax is bounded by $\Omega((\log n)/\rho)$, too. It is an open problem to show matching upper and lower bounds. We conjecture that the lower bound for OneMax is far from optimal and that $\Omega(n/\rho + n \log n)$ holds.

4.3 LeadingOnes

Another prominent unimodal example function, proposed by Rudolph (1997), is

$$\textsc{LeadingOnes}(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_j,$$

whose function value equals the number of leading ones in the considered bit string x. A non-optimal solution may always be improved by appending a single one to the leading ones. LeadingOnes differs from OneMax in the essential way that the assignment of the bits after the leading ones do not contribute to the function value. This implies that bits at the beginning of the bit string have a stronger influence on the function value than bits at the end. Because of this, the methods developed by Neumann and Witt (2006) cannot be used for analyzing the 1-ANT on LeadingOnes as these methods make particular use of the fact that all bits contribute equally to the function value. In a follow-up paper by Doerr, Neumann, Sudholt, and Witt (2007), the 1-ANT is studied on LeadingOnes by different techniques and it is shown that a similar phase transition behavior as on OneMax exists: for $\rho = o(1/\log n)$ (again using the notation of the present paper), the expected optimization time of the 1-ANT is superpolynomially large whereas it is polynomial for $\rho = \Omega(1/\log n)$ and even only $O(n^2)$ for $\rho = \Omega(1)$. We already know that this phase transition cannot occur with MMAS* and MMAS on LeadingOnes. The following theorem,

special cases of which are contained in Gutjahr and Sebastiani (2007), shows a specific upper bound for MMAS*.

Theorem 9 *The expected optimization time of MMAS* on* LeadingOnes *is bounded from above by* $O((n \log n)/\rho + n^2)$.

Proof: The theorem follows again by the bound (2). We use the same fitness-based partition as in the proof of Theorem 7 and the expected optimization time $O(n^2)$ of the (1+1) EA on LeadingOnes (Droste, Jansen and Wegener, 2002). □

It is interesting that an almost tight lower bound can be derived. The following theorem shows that the expected optimization time of MMAS* is never better than $\Omega(n^2)$. The proof is lengthy, however, for the case of large ρ, one essential idea is easy to grasp: already in the early stages of the optimization process, many, more precisely $\Omega(n)$, pheromone values on 1-edges reach their lower borders $1/n$, and the corresponding bits are set to 0. To "flip" such a bit, events of probability $1/n$ are necessary. This can be transformed into the lower bound $\Omega(n^2)$ on the expected optimization time.

Theorem 10 *Choosing* $\rho = 1/\text{poly}(n)$*, the expected optimization time of MMAS* on* LeadingOnes *is bounded from below by* $\Omega(n/\rho + n^2)$.

Proof: We first show the lower bound $\Omega(n/\rho)$ on the expected optimization time. Afterwards, this bound is used to prove the second bound $\Omega(n^2)$. Throughout the proof, we consider only runs of polynomial length since by the assumption $\rho = 1/\text{poly}(n)$ the lower bounds to show are both polynomial. This allows us to ignore events with exponentially small probabilities.

For the first part, we use the observation by Doerr, Neumann, Sudholt, and Witt (2007) on the pheromone values outside the block of leadings ones. If the best-so-far LeadingOnes-value equals k, the pheromone values corresponding to the bits $k+2, \ldots, n$ have never contributed to the LO-value implying that each of these bits is unbiasedly set to 1 with probability exactly $1/2$ in the next constructed solution. Note that these pheromone values are not necessarily martingales, however, the probability distribution of such a pheromone value is symmetric with expectation $1/2$. Hence we distinguish two states for a bit: if it is right of the leftmost zero in the best-so-far solution, its expected pheromone values on 1- and 0-edges equal $1/2$; if it is left of the leftmost zero, the pheromone value of its 1-edges are monotonically increasing in each iteration until the border $1 - 1/n$ is reached. We call the first state the *random state* and the second one the *increasing state*.

While all bits in the block $n/4+1, \ldots, n/2$ are in random state, the probability of obtaining a LeadingOnes-value of at least $n/2$ is bounded from above by $2^{-\Omega(n)}$. Hence, with probability $1 - 2^{-\Omega(n)}$, the LeadingOnes-value is in the interval $[n/4, n/2]$ at some point of time. The randomness of the bits after the leftmost zero allows us to apply the standard free-rider arguments by Droste,

Jansen and Wegener (2002). Hence, with probability $1 - 2^{-\Omega(n)}$, at least $n/12$ improvements of the LeadingOnes-value have occurred when it has entered the interval $[n/4 + 1, n/2]$. This already completes the first part of the proof if $\rho = \Omega(1)$. In the following, we therefore study the case $\rho = o(1)$. Assume for some arbitrarily slowly increasing function $\alpha(n) = \omega(1)$ that only $n/(\alpha(n)^2\rho)$ iterations have happened until the first $n/12$ improvements are done. Then it must hold (by the pigeon-hole principle) that at most $n/\alpha(n)$ times, the number of iterations between two consecutive improvements is large, which is defined as: *not* bounded by $O(1/(\alpha(n)\rho))$. Furthermore, by another pigeon-hole-principle argument, there must be at least $n/\alpha(n)$ independent so-called fast phases defined as follows: each fast phase consists of at least $r := \lfloor \alpha(n)/24 \rfloor$ consecutive improvements between which the number of iterations is never large, i. e., each time bounded by $O(1/(\alpha(n)\rho))$. (For a proof, consider a fixed subdivision of $n/12$ improvements into blocks of r consecutive improvements and show that at most half of these blocks can contain a large number of iterations between some two consecutive improvements.) In the following, we will show that a fast phase has probability $o(1)$. This contradicts (up to failure probabilities of $2^{-\Omega(n/\alpha(n))}$) the assumption we have at least $\Omega(n/\alpha(n))$ fast phases, hence the number of iterations for the first $n/12$ improvements cannot be bounded by $n/(\alpha(n)^2\rho)$. Since $\alpha(n)$ can be made arbitrarily slowly increasing, we obtain the lower bound $\Omega(n/\rho)$ on the expected optimization time.

Consider the event that a fast phase containing r improvements is sufficient to set at least r bits into the increasing state (the phase is called successful then). In the beginning, all these bits are in random state. Hence, with a failure probability at most $2^{-\Omega(r)}$, less than $r/4$ pheromone values on the corresponding 1-edges (in the following called success probabilities) are at most $1/2$. We assume $r/4$ bits with this property and estimate the probability of setting all these bits to 1 simultaneously in at least one improving step until the end of the phase (which is necessary for the phase to be successful). The success probability of a bit with initial pheromone value $1/2$ is still at most $p_t := 1 - (1-\rho)^t/2$ if it has been only in increasing state for t steps. The total number of iterations in the phase is $O(1/\rho)$ by the definition of a fast phase. Hence, by the end of the phase, all considered success probabilities are at most $1 - (1-\rho)^{O(1/\rho)}/2 = 1 - \Omega(1)$. The probability of a single improving step setting the $r/4$ bits to 1 is therefore at most $(1 - \Omega(1))^{r/4} = 2^{-\Omega(r)}$. Adding up over all r improving steps and taking into account the above failure probability, the probability of the phase being successful is at most $r2^{-\Omega(r)} + 2^{-\Omega(r)} = 2^{-\Omega(\alpha(n))} = o(1)$ as suggested.

Having proved that the expected number of steps for $n/12$ improvements is $\Omega(n/\rho)$, we conclude that there is a constant $c > 0$ such that the time between two improvements is at least c/ρ with probability at least $1/2$. Otherwise Chernoff bounds would (up to exponentially small failure probabilities) contradict the bound $\Omega(n/\rho)$. We exploit this to show that with high probability, a linear number of bits in random state reaches the lower border $1/n$ on the success probability during the optimization process. This will prove the second lower bound $\Omega(n^2)$ on the expected optimization time.

Consider a bit in random state and a phase of $t^* \leq (\ln n)/\rho$ iterations. We are interested in the event that the bit is set to zero throughout all improvements of the phase, which implies that the success probability is $1/n$ until the end of the phase (the phase is finished prematurely if the border $1/n$ is reached in less than t^* steps). This event has probability $\Omega(1)$ for the following reasons: let p_0 be the initial probability of setting the bit to zero and assume $p_0 \geq 1/2$ (which holds with probability at least $1/2$). After t steps of the phase, this probability is at least $p_t := 1 - (1-\rho)^t/2$ if the bit has only been set to zero in the improvements in the phase. If the time between two improvements (and, therefore, exchanges of the best-so-far solution) is always bounded from below by c/ρ, the considered event has probability at least $\prod_{t=1}^{\infty} p_{tc/\rho}$. Using $1-x \leq e^{-x}$ for $x \in \mathbb{R}$ and $1-x \geq e^{-2x}$ for $x \leq 1/2$, this term can be bounded from below by

$$\prod_{t=1}^{\infty}\left(1-\frac{(1-\rho)^{tc/\rho}}{2}\right) \geq \prod_{t=1}^{\infty}\left(1-\frac{e^{-ct}}{2}\right) \geq \prod_{t=1}^{\infty}\exp(e^{-ct})$$
$$= \exp\left(\sum_{t=1}^{\infty} e^{-ct}\right) = \exp\left(\frac{e^{-c}}{1-e^{-c}}\right) = \Omega(1).$$

That the phase is of the desired length only with probability $1/2$ is no problem either since we can bound the probability of setting the bit to 0 after a short phase by the probability in the beginning of the phase and argue like in the proof of Theorem 2 by Doerr, Neumann, Sudholt, and Witt (2007). The event of a short phase corresponds to the occurrence of a "free-rider" and the number of short phases is geometrically distributed with success probability at most $1/2$. Hence, using the same calculations as in the mentioned proof, the probability of the desired event is at least $\prod_{t=1}^{\infty} \frac{p_{tc/\rho}}{2-p_{tc/\rho}} \geq \prod_{t=1}^{\infty} \frac{p_{tc/\rho}}{2} = \Omega(1)$ as claimed.

Using the observation from the last paragraph, it follows that with probability $1-2^{-\Omega(n)}$, at least $\Omega(n)$ bits in random state have reached success probability $1/n$ by the time the LeadingOnes-value enters the interval $[n/4, n/2]$. The only problem might be that these success probabilities might increase again. However, in each of the remaining improvements, we distinguish the events whether it is relevant for an improvement to set such a bit to 1 or not. If the bit is not relevant since it is still right of the leftmost zero after the improvement, its success probability does not change with probability $1-1/n$. Hence, in $O(n)$ improvements there are in expectation still $\Omega(n)$ success probabilities equal to $1/n$ left. Since, with at least constant probability, $\Omega(n)$ improvements are necessary, the lower bound $\Omega(n^2)$ on the expected optimization time follows. □

In the proof, we had to carefully look at the random bits and to study the structure of the optimization process. It seems to be even harder to prove a corresponding lower bound for MMAS since accepting equally good solutions implies that more than n exchanges of the best-so-far solution can happen. Also additional ideas are required to transfer the proof of Theorem 10 and to obtain an improved lower bound for MMAS* on OneMax.

5 Conclusions

The rigorous runtime analysis of ACO algorithms is a challenging task where the first results have been obtained only recently. In this paper, we have considered an ACO algorithm called MMAS for which some results based on the method of fitness-based partitions have been obtained. Previous results on this algorithm by Gutjahr and Sebastiani have been extended and improved and compared to our earlier findings for the 1-ANT. In particular, we have considered some unimodal functions such as OneMax and LeadingOnes and proved upper and lower bounds. Furthermore, we have argued why it is necessary to replace search points by other ones that have the same fitness and shown that this improves the runtime on the well-known plateau function Needle.

References

Doerr, B., Neumann, F., Sudholt, D., and Witt, C. (2007). On the runtime analysis of the 1-ANT ACO algorithm. In *Proc. of GECCO '07*. ACM. (to appear).

Dorigo, M. and Blum, C. (2005). Ant colony optimization theory: A survey. *Theor. Comput. Sci.*, **344**, 243–278.

Dorigo, M., Maniezzo, V., and Colorni, A. (1991). The ant system: An autocatalytic optimizing process. Tech. Rep. 91-016 Revised, Politecnico di Milano.

Dorigo, M. and Stützle, T. (2004). *Ant Colony Optimization*. MIT Press.

Droste, S., Jansen, T., and Wegener, I. (2002). On the analysis of the (1+1) evolutionary algorithm. *Theor. Comput. Sci.*, **276**, 51–81.

Garnier, J., Kallel, L., and Schoenauer, M. (1999). Rigorous hitting times for binary mutations. *Evolut. Comput.*, **7**(2), 173–203.

Gutjahr, W. J. (2002). ACO algorithms with guaranteed convergence to the optimal solution. *Inform. Process. Lett.*, **82**, 145–153.

Gutjahr, W. J. (2006). On the finite-time dynamics of ant colony optimization. *Methodol. Comput. Appli. Probab.*, **8**, 105–133.

Gutjahr, W. J. (2007a). First steps to the runtime complexity analysis of Ant Colony Optimization. *Comput. Oper. Res.* (to appear).

Gutjahr, W. J. (2007b). Mathematical runtime analysis of ACO algorithms: Survey on an emerging issue. *Swarm Intelligence*. (to appear).

Gutjahr, W. J. and Sebastiani, G. (2007). Runtime analysis of ant colony optimization. Tech. rep., Mathematics department, "Sapienza" Univ. of Rome, 2007/03.

Jansen, T. and Wegener, I. (2001). Evolutionary algorithms - how to cope with plateaus of constant fitness and when to reject strings of the same fitness. *IEEE Trans. Evolut. Comput.*, **5**(6), 589–599.

Mitzenmacher, M. and Upfal, E. (2005). *Probability and Computing – Randomized Algorithms and Probabilistic Analysis*. Cambr. Univ. Press.

Motwani, R. and Raghavan, P. (1995). *Randomized Algorithms*. Cambr. Univ. Press.

Neumann, F. and Witt, C. (2006). Runtime analysis of a simple ant colony optimization algorithm. In *Proc. of ISAAC '06*, vol. 4288 of *LNCS*, 618–627. Springer. Extended version to appear in *Algorithmica*.

Rudolph, G. (1997). *Convergence Properties of Evolutionary Algorithms*. Kovač.

Stützle, T. and Hoos, H. H. (2000). MAX-MIN ant system. *J. Future Gener. Comput. Syst.*, **16**, 889–914.

Wegener, I. (2002). Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. In Sarker, R., Yao, X., and Mohammadian, M. (eds.), *Evolutionary Optimization*, 349–369. Kluwer.

Wegener, I. and Witt, C. (2005). On the optimization of monotone polynomials by simple randomized search heuristics. *Combin. Probab. Comput.*, **14**, 225–247.