How Single Ant ACO Systems Optimize Pseudo-Boolean Functions*

Benjamin Doerr, Daniel Johannsen, Ching Hoo Tang

Max-Planck-Institut für Informatik
Campus E1 4, 66123 Saarbrücken, Germany

Abstract

We undertake a rigorous experimental analysis of the optimization behavior of the two most studied single ant ACO systems on several pseudo-boolean functions. By tracking the behavior of the underlying random processes rather than just regarding the resulting optimization time, we gain additional insight into these systems. A main finding is that in those cases where the single ant ACO system performs well, it basically simulates the much simpler (1+1) evolutionary algorithm.

1 Introduction

In 1991, Dorigo, Maniezzo and Colorni [4] introduced the concept of Ant Colony Optimization (ACO). Since then ACO algorithms have been applied successfully to many kinds of combinatorial problems, e. g., the famous Travelling Salesman Problem. See the book by Dorigo and Stützle [5] and the references therein.

In the last few years, theoretical research has been started to gain an understanding of why these methods are so successful. Since the probability space describing a run of a typical ACO system is extremely complicated, theoretical works concentrated on the runtime behavior of two ACO systems involving a single ant only, namely 1–Ant and the Max–Min Ant System (MMAS).

The algorithm 1–Ant was proposed by Neumann and Witt [11]. It is an adaption of the more general Graph–Based Ant System introduced by Gutjahr [7] to allow optimizing (non-graph based) pseudo-boolean functions. Gutjahr and Sebastiani [8] gave the first rigorous runtime analysis of the MMAS and showed that on certain needle-in-a-haystack functions their ACO system beats the classical (1+1) evolutionary algorithm ((1+1) EA) (for both algorithms a version was used that does not accept new solutions of equal fitness). Neumann and Witt [11] conducted the runtime analysis of 1–Ant on the pseudo-boolean function OneMax (counting the number of ones in an n–bit string). While obviously a highly simplified problem, the analysis was far from simple. The main outcome of the analysis is that the optimization time depends crucially on the major parameter, the so-called evaporation factor ρ. Here, crucially means that there is a relatively sharp distinction between quite efficient optimization and exponential run-time behavior. Additionally, it was observed that for ρ very close to one, 1–Ant exactly simulates the (1+1) EA, which was rigorously analyzed in [6]. In [2] the pheromone model used in [11] was replaced by a simpler, but equivalent model, in which the pheromone values equal the probabilities of ants walking along a particular edge in the construction graph. In [3] an analysis of 1–Ant was carried out for the functions LeadingOnes and BinaryValue. As in the OneMax case, a phase transition from exponential to polynomial runtime with a threshold evaporation value could be observed.

*This is the author's version of the work. It is posted here for your personal usen not for redistribution. The definitive version was published in *Proceedings of Parallel Problem Solving from Nature — PPSN X*, 2008, pages 378–388, LNCS. It is available at http://www.springerlink.com/content/l0311j2v27vk748g.

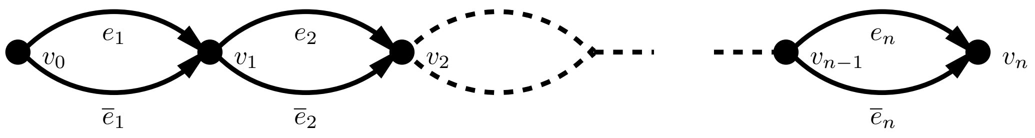

Figure 1: Bit-strings are represented by ant walks on the simplified chain graph.

The algorithm MMAS by Stützle and Hoos [12] was studied by Gutjahr and Sebastiani [8] and Neumann, Sudholt and Witt [9]. Roughly speaking, these works show that several variants of the MMAS less critically depend on the choice of the parameter ρ in the sense that there is no sharp phase transition between polynomial and exponential runtime as observed with 1–Ant. More recently, the benefits and shortcomings of hybridizations of the MMAS with local search strategies have been investigated [10].

Some of the run-time analyses sketched above could be read as that single ant ACO system are competitive approaches to optimize pseudo-boolean functions. To further study this aspect, we conduct a rigorous experimental analysis of these two single ant ACO systems on several pseudo-boolean fitness functions (OneMax, LeadingOnes, and linear functions with random weights). To gain an understanding how these algorithms work, we track a number of theory–guided indicators (other than the resulting optimization time) during the runs of 1–Ant [11] and the MMAS [12]. For both algorithms we use the pheromone system described in [2] which is equivalent to the one used in [8], [11], and [3].

Our main finding is that whenever one of the two ACO systems for a certain choice of ρ has an at least roughly reasonable run-time, then its optimization behavior is very similar to that of the (1+1) EA. This shows that the pessimistic assumptions repeatedly used in the proofs of the results mentioned above are real, and in consequence, indicates that the upper bounds on the optimization time proven there probably cannot be improved. Our analysis of the optimization behavior fits well to the fact that we rarely observe that one of the two single ant ACO systems finds the optimum significantly faster than the (1+1) EA [1].

2 Single Ant ACO and the (1+1) EA

Given a *fitness function* $f\colon \{0,1\}^n \to \mathbb{R}$ on the bit-strings of length n, a *single ant ACO algorithm* successively generates candidate solutions $S^{(t)} \in \{0,1\}^n$ according to the *pheromone values* $p^{(t)} \in [0,1]^n$. We understand this sampling process as a random walk of a single ant on the directed *construction graph* depicted in Figure 1. At each vertex v_{i-1} the ant chooses one of the two outgoing edges e_i or $\overline{e}_i$ with probability equal to the *pheromone value* $p_i^{(t)}$ or $1 - p_i^{(t)}$, respectively. If the ant chooses e_i, we have $S_i^{(t)} = 1$ and $S_i^{(t)} = 0$ otherwise.

```
**AntWalk**($p$)
1  for $i \in \{1,\dots,n\}$ do choose $S_i \in \{0,1\}$ with $\Pr(S_i = 1) = p_i$
2  return $S$
```

Initially, all pheromone values are 1/2. Hence, the first ant performs a true random walk. Later, updates to the pheromone values are triggered by certain ant walks. In this case, a certain amount of pheromone evaporates from all edges and then the pheromone values of the edges the ant traverses are reinforced. The amount of both, evaporation and reinforcement, is governed by the algorithm's main parameter, the *evaporation factor* $\rho \in [0,1]$.

```
Update($p, S, \rho$)
1  for $i \in \{1,\dots,n\}$ do
2      if $S_i = 1$ then $p'_i := \min\{(1-\rho)\cdot p_i + \rho, 1-\frac{1}{n}\}$ else $p'_i := \max\{(1-\rho)\cdot p_i, \frac{1}{n}\}$
3  endfor
4  return $p'$
```

In this theoretical investigation, we run both algorithms for a number of $t_{\max} \in \mathbb{N}$ generations and then return the best solution found so far. In practice, other stopping criteria might be more appropriate.

```
1–Ant ($f, t_{\max}, \rho$)
 1  $p^{(0)} := (1/2, ..., 1/2)$
 2  $S_{\max} :=$ AntWalk($p^{(0)}$)
 3  $p^{(1)} :=$ Update($p^{(0)}, S_{\max}, \rho$)
 4  for $t$ from 1 to $t_{\max}$ do
 5      $S^{(t)} :=$ AntWalk($p^{(t)}$)
 6      if $f(S^{(t)}) \geq f(S_{\max})$ then
 7          $S_{\max} := S^{(t)}$
 8          $p^{(t+1)} :=$ Update($p^{(t)}, S_{\max}, \rho$)
 9      endif
10  endfor
11  return $S_{\max}$
```

```
MMAS ($f, t_{\max}, \rho$)
 1  $p^{(0)} := (1/2, ..., 1/2)$
 2  $S_{\max} :=$ AntWalk($p^{(0)}$)
 3  $p^{(1)} :=$ Update($p^{(0)}, S_{\max}, \rho$)
 4  for $t$ from 1 to $t_{\max}$ do
 5      $S^{(t)} :=$ AntWalk($p^{(t)}$)
 6      if $f(S^{(t)}) \geq f(S_{\max})$ then
 7          $S_{\max} := S^{(t)}$
 8      endif
 9      $p^{(t+1)} :=$ Update($p^{(t)}, S_{\max}, \rho$)
10  endfor
11  return $S_{\max}$
```

1–Ant and the MMAS both simulate one of two well–known randomized search heuristics if the evaporation factor is zero or one. For $\rho = 0$, they simply perform random search, and for $\rho = 1$, they precisely simulate the (1+1) EA with mutation probability $1/n$.

3 The Experimental Setup

The classical mean to measure the performance of a randomized search heuristic is the *optimization time* T, which is the number of fitness evaluations needed to find the optimal solution. For efficiency reasons, we introduce an artificial upper bound of $t_{\max} = 1000000$, i. e., $T = \min\{t \in \mathbb{N} \mid f(S^{(t)})$ is optimal or $t = t_{\max}\}$.

To analyze the optimization behavior, we monitor a number of theory–guided indicators measuring the algorithm's progress at every single step $t \in \mathbb{N}$ of the run. In particular, we investigate the fitness of the current solution $f(S^{(t)})$, its expectation $\mu^{(t)} := \mathrm{E}[f(S^{(t)})]$ and variance $\nu^{(t)} := \mathrm{Var}[f(S^{(t)})]$, the fitness of the best solutions so far $f^{(t)}_{\max} := \max_{r \leq t} f(S^{(r)})$, the number $mm := |\{i \mid p^{(t)}_i \in \{1/n, 1-1/n\}\}|$ of pheromone values attaining one of the boundary values $1/n$ and $1-1/n$, and the probability $P^{(t)} := \Pr(f(S^{(t)}) \geq f^{(t)}_{\max})$ of accepting the current solution.

We also investigate the average values $\overline{P}$, $\overline{\nu}$, and $\overline{mm}$ of $P^{(t)}$, $\nu^{(t)}$, and $mm^{(t)}$ over the interval $[0.25\,T, 0.75\,T]$. This interval was chosen to eliminate possible side-effects at the beginning and the end of a run.

We study the progress behavior of the two ant optimization algorithms 1–Ant and the MMAS on different pseudo-boolean fitness functions $f\colon \{0,1\}^n \to \mathbb{R}$. We regard *random linear functions* $f(S) = \sum_{i=1}^{n} w_i S_i$, where the weights $w_1, \dots, w_n$ are chosen independently and uniformly at random in $(0, 1]$, and then normalized to add up to n. Clearly, the normalization does not change the behavior of any of the algorithms, but eases comparing the results for different functions. Furthermore, we regard the two classical pseudo-boolean test functions $\textsc{OneMax}(S) = \sum_{i=1}^{n} S_i$ and $\textsc{LeadingOnes}(S) = \sum_{k=1}^{n} \prod_{i=1}^{k} S_i$. Clearly, $S^* = (1, \dots, 1)$ is the unique maximum of all these functions having fitness $f(S^*) = n$. In the experiments, we use a problem size of $n = 1000$ for OneMax and random linear functions, and a problem size of $n = 200$ for LeadingOnes.

Also, note that for LeadingOnes, $\mu^{(t)}$ and $\nu^{(t)}$ as defined above are heavily influenced by the fact that with probability around $1/e$, one of the leading one–bits (having pheromone value $1-1/n$) will be set to zero. Since such a solution will not be accepted anyway, for LeadingOnes we modify the definitions of $\mu^{(t)}$ and $\nu^{(t)}$ to be the expectation and variance conditional on that none of these leading bits is zero.

We omit the details on how to actually compute the progress indicators for these test functions. In all but one case, this can be done efficiently in linear time or via dynamic programming in quadratic time. For arbitrary linear function, however, $P^{(t)}$ cannot be computed efficiently. In consequence, we cannot provide $P^{(t)}$ and $\overline{P}$ for random linear functions.

We perform case studies to analyze the time–dependent indicators $f(S^{(t)})$, $f_{\max}^{(t)}$, $\mu^{(t)}$, $\nu^{(t)}$, $P^{(t)}$, and $mm^{(t)}$. That is, for all algorithms and test functions, we conduct twenty runs each for at least twenty different ρ-values, graphically depict the indicators and single out typical runs for representative values of ρ. In all cases, we see that the indicators for random linear functions and OneMax behave highly similar. For this reason, in the following two sections we present and discuss plots for OneMax only, since here we also have the indicator $P^{(t)}$.

To measure the average indicators $\overline{\nu}$, $\overline{P}$, and $\overline{mm}$, we performed 20 runs for both algorithms on all fitness functions and several values of ρ. We then average the average indicators over all runs. Since the success of 1–Ant depends sharply on ρ, we happen to never average over successful and unsuccessful runs simultaneously. For unsuccessful runs we also record the final values of $f_{\max}$ and P. For reasons of space, we can only present a tiny fraction of the data collected. Much additional material can be accessed in [1].

4 Experimental Results for 1–Ant

In this section, we present our experimental work concerning the single ant ACO system 1–Ant. In [11] it was shown that the expected optimization time of 1–Ant on OneMax is polynomial if $\rho = 1 - 1/n^{\epsilon}$ with fixed $\epsilon > 0$ and in [2] that for $\rho = o(1/\log n)$ it becomes super–polynomial. On the function LeadingOnes the expected optimization time was shown [3] to be quadratic for constant ρ, polynomial for $\rho = \Omega(1/\log n)$, and again super-polynomial for $\rho = o(1/\log n)$. Note that $\rho = 2n\tilde{\rho}/(1-\tilde{\rho}+2n\tilde{\rho})$ for the evaporation factor $\tilde{\rho}$ in [8], [11] and [3].

Since we argue that for efficient runs of 1–Ant the optimization behavior strongly resembles that of

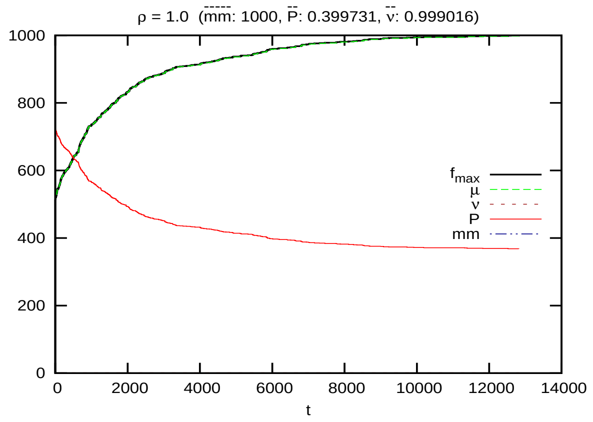

Figure 2: A typical run of the (1+1) EA on OneMax. Since $\nu^{(t)} = 1 - 1/n$ and $mm^{(t)} = n$ for all t, these curves coincide with the lower and upper boundaries of the chart. Also, $f_{\max}^{(t)}$ and $\mu^{(t)}$ are too close to each other to be distinguished. In the chart we rescale $p^{(t)}$ from $[0, 1]$ to $[0, 1000]$.

the (1+1) EA, let us first present a typical run for the (1+1) EA. Note that 1–Ant for $\rho = 1$ exactly simulates the (1+1) EA, so we can reuse our test environment here. Figure 2 shows such a typical run as a chart of the indicators $f_{\max}^{(t)}$, $\mu^{(t)}$, $\nu^{(t)}$, $P^{(t)}$, and $mm^{(t)}$. As it is easy to see, the (1+1) EA improves the fitness relatively fast. This is natural, since the probability $P^{(t)}$ of finding an acceptable solution remains very large during the whole run. For the same reason, $f_{\max}^{(t)}$ and $\mu^{(t)}$ are that close together.

We now analyze one representative run of the algorithm 1–Ant on OneMax for each of the pheromone values $\rho = 0.4$, 0.2, 0.1, and 0.05, which give a good overview of the different behaviors of 1–Ant. The plots are depicted in Figure 3.

For $\rho = 0.4$, we easily identify a behavior highly similar to that of the (1+1) EA. In a very short initial phase of approximately 130 iterations, almost all pheromone values are pushed to their extreme values and the variance drops from its initial value of $n/4$ to a value close to one. Note that $f_{\max}$ remains close to $n/2$ in this initial phase. In consequence, this means that half of the pheromone values attain the maximum value and half the minimum. By symmetry, they are randomly chosen as is the initial solution in the (1+1) EA. From this point on, the optimization behavior of the 1–Ant closely resembles the one of the (1+1) EA. This is easily seen from the curves and the average values $\overline{P}$, $\overline{mm}$ and $\overline{\nu}$. In the chart it seems that $P^{(t)}$ oscillates heavily, but as the average $\overline{P}$ indicates, these downward dents are only short-term occurrences. They stem mainly from the fact that after an improvement in the fitness, the pheromone values affected take a few iterations until they hit the extreme values again.

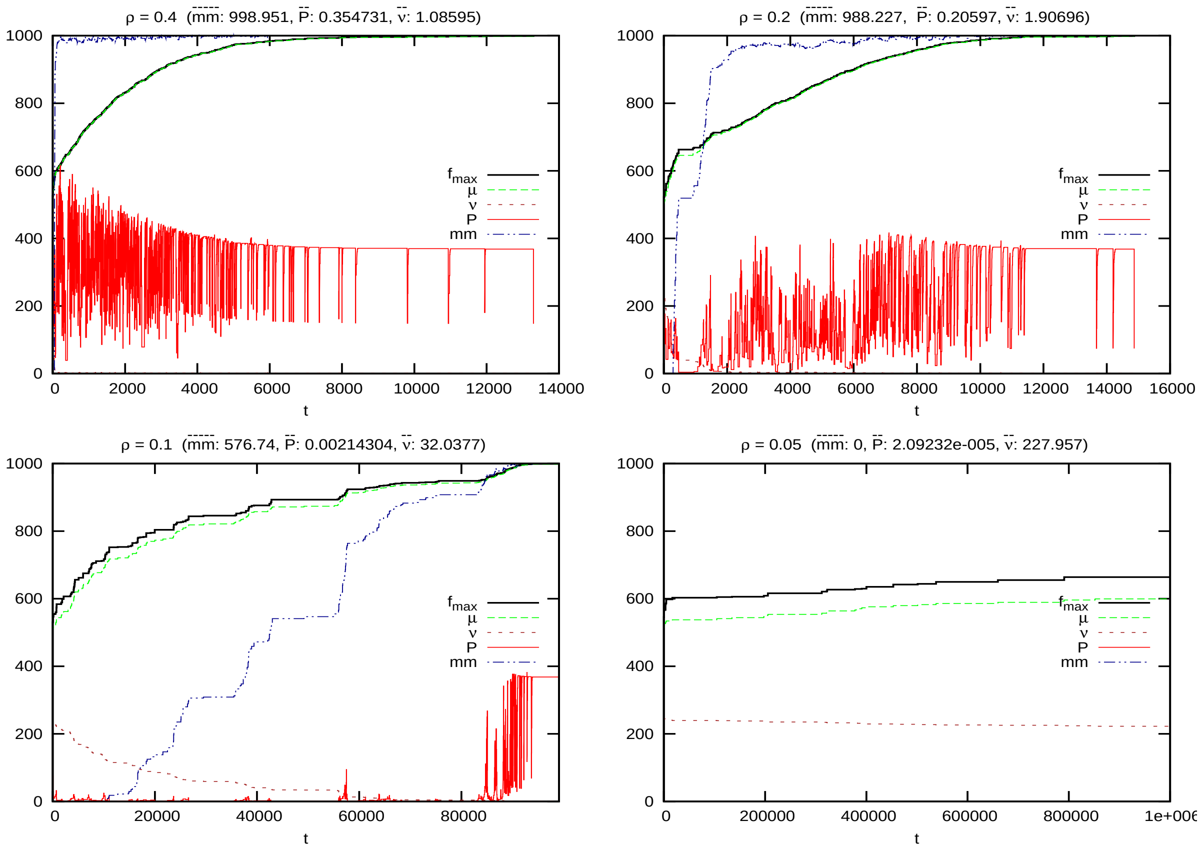

Figure 3: Four typical runs of 1–Ant on OneMax for $\rho = 0.4$ (top left), 0.2 (top right), 0.1 (bottom left), and 0.05 (bottom right).

The chart for $\rho = 0.2$ still shows many signs of an (1+1) EA-like behavior. However, we also see first short phases of stagnation. At $t = 467$, a solution of value 663 is found (+11 to the previous best). Due to the slower pheromone update with $\rho = 0.2$, this increases the expected value of the next solution only from 642 to 646, and spoils the probability of finding an acceptable solution down to a mere 0.4%. In consequence, it takes the 1–Ant a long 450 iterations to find an as good solution again (at time $t = 917$).

For $\rho = 0.1$, the phenomenon just described becomes much more dominant. We now have several long phases in which no solution is accepted. Finally, for $\rho = 0.05$ this pattern is so strong that no solution is found within one million iterations. With $\overline{P}$ around $2 \cdot 10^{-5}$, it is very hard to find an acceptable solution. Recall that $P^{(t)} = 2 \cdot 10^{-5}$ means that an expected number of 50.000 iterations are necessary to generate a solution that is accepted.

What we just extracted from the charts in Figure 3 is also visible from Table 1. For both, random linear functions and OneMax, we observe the expected behavior. For $\rho \geq 0.2$, we have a (1+1) EA-like optimization behavior. All but very few pheromone values are at their extreme values. Those who are not, still are very close to the extreme values, as can be deduced from the variance. For $\rho \leq 0.1$, finding acceptable solutions becomes increasingly hard. $\overline{P}$ values of 10^{-3} or less indicate that fewer than every thousandth solution generated is actually accepted.

From $\rho \leq 0.06$ on ($\rho \leq 0.08$ for linear functions), as few solutions are accepted that (a) one million runs never sufficed to find the optimum, (b) the variance is close to the maximum value of 250 (approx. 330 for random linear functions), indicating that most pheromone values are close to their initial values. To add a number, averaging over 20 runs with $\rho = 0.05$ we found that less than 49 (of the one million generated) solutions are accepted.

We see very similar results if we use LeadingOnes as fitness function. For reasons of space, we only present the data in table form (Table 2). Again, we see that for $\rho \geq 0.2$, most pheromone values are at their extreme values and the optimization time is not much different from the case $\rho = 1$, which again is the (1+1) EA. The phase transition happens for slightly smaller ρ values. Up to $\rho = 0.08$, we see still reasonable optimization times. From then on, however, the optimization time increases again drastically and $\overline{P}$ falls to ridiculously small values. For $\rho \leq 0.045$, no run is successful, and final $P^{(t)}$ values in the 10^{-5} to 10^{-6} range show that some 100.000 iterations are necessary to find an acceptable solution (which not necessarily leads to an increased fitness).

	OneMax						linear functions			
ρ	T	$\overline{mm}$	$\overline{P}$	$\overline{\nu}$	$f_{\max}^{(t_{\max})}$	$P^{(t_{\max})}$	T	$\overline{mm}$	$\overline{\nu}$	$f_{\max}^{(t_{\max})}$
1.0	18246	1000	0.38	0.999			17773	1000	1.332	
0.5	10953	998.8	0.36	1.104			16756	999.6	1.350	
0.1	146895	352.4	$1.82 \cdot 10^{-3}$	66.038			579284	414.6	52.153	
0.05	1000000	0	$1.82 \cdot 10^{-5}$	213.223	691	$7.68 \cdot 10^{-6}$	1000000	0	298.104	673
0.01	1000000	0	$3.96 \cdot 10^{-6}$	248.732	582	$1.95 \cdot 10^{-6}$	1000000	0	332.857	595

Table 1: Results for 1–Ant optimizing the fitness functions OneMax and random linear functions. All numbers are the averages over 20 runs. For $\rho = 0.05$ and $\rho = 0.01$, where no run is successful, we also list the average optimum and acceptance probability at $t_{\max} = 1000000$.

ρ	T	$\overline{mm}$	$\overline{P}$	$\overline{\nu}$	$f_{\max}^{(T)}$	$P^{(T)}$
1.0	34768	200	0.55	0.03		
0.5	33964	197	0.54	0.06		
0.1	35175	177	0.47	0.60		
0.01	1000000	0	$4.47 \cdot 10^{-6}$	3.33	23	$1.90 \cdot 10^{-6}$
0.005	1000000	0	$3.52 \cdot 10^{-6}$	2.49	22	$1.64 \cdot 10^{-6}$
0.001	1000000	0	$3.20 \cdot 10^{-6}$	2.09	21	$1.09 \cdot 10^{-6}$

Table 2: Indicators for the behavior of 1–Ant optimizing LeadingOnes for different values of ρ. For $\rho = 1.0$, 0.5, and 0.1 all runs find the optimum in at most 10^6 steps, for $\rho = 0.01$, 0.005, and 0.001 none. We omit $\rho = 0.05$ to avoid the bias caused by T not reaching 10^6 in all of the runs.

5 Experimental Results for the MMAS

We now analyze the optimization behavior observed for the MMAS. In [8] and [9], expected optimization times of $O(\rho^{-1} n \log(n))$ and $O(n^2 + \rho^{-1} n \log n)$ were proven for a variant MMAS on the two pseudo-boolean functions OneMax and LeadingOnes. This indicates that for the MMAS the optimization time does not display such a delicate dependence on ρ as previously seen for 1–Ant. Our experimental results verify this observation, but again show that the MMAS strongly imitates the optimization behavior of the (1+1) EA, in particular for the more efficient runs with larger ρ.

Our experimental results on the optimization behavior of the MMAS for linear functions is summarized in Figure 4 and Table 3. For random linear functions and OneMax, all runs, even those for relatively small ρ values like 0.005, show an optimization behavior strongly resembling that of the (1+1) EA. The average number $\overline{mm}$ of pheromone values having one of the two extremal values is above 970. In other words, in average at least 97% of the bits of the newly generated solution are determined in the same way as by the (1+1) EA. The remaining pheromone values are also close to their extreme values, as witnessed by an average variance $\overline{\nu}$ of less than 3. Not surprisingly, the optimization times are similar to the (1+1) EA-case with a considerable slow-down for small ρ values.

In Figure 4, we depict four typical runs for OneMax. We immediately notice that the four graphs are much more similar than those for 1–Ant in Figure 3, even though a wider range of ρ–values is covered.

For $\rho = 0.4$, 0.25 and 0.1 we observe an extremely short initial phase during which the pheromone values

	OneMax				linear functions					LeadingOnes			
ρ	T	$\overline{mm}$	$\overline{P}$	$\overline{\nu}$	T	$\overline{mm}$	$\overline{\nu}$		ρ	T	$\overline{mm}$	$\overline{P}$	$\overline{\nu}$
1.0	16151	1000	0.38	0.999	18399	1000	1.336		1.0	34749	200	0.54	0.03
0.5	13325	999.7	0.38	1.021	17175	999.9	1.324		0.5	33481	198	0.56	0.05
0.1	11972	997.1	0.33	1.184	17454	998.9	1.356		0.1	32759	188	0.54	0.11
0.05	14054	994.2	0.28	1.344	19447	997.1	1.428		0.05	32714	181	0.52	0.23
0.01	22504	980.0	0.12	2.159	30966	987.9	1.813		0.01	32003	159	0.41	1.28
0.005	30398	971.2	0.07	2.667	44426	983.6	1.957		0.005	35719	152	0.34	2.31
0.001	75076	952.5	0.02	3.757	107027	970.2	2.509		0.001	66000	138	0.15	12.07

Table 3: Indicators for the behavior of MMAS optimizing OneMax and random linear functions ($n = 1000$), as well as LeadingOnes ($n = 200$) for different ρ values.

rush towards their extreme values. After 14, 37, and 114 steps, respectively, 95% of the pheromone values are $1/n$ or $1-1/n$. After this initial phase, in which $f_{\max}^{(t)}$ does not exceed 527 (559, 562, respectively), all indicators are similar to what we see for the (1+1) EA. This is obvious for $mm^{(t)}$, which is close to $n = 1000$ all the time, and ν, which is too small to be distinguished from the t–axis. $P^{(t)}$ differs from the (1+1) EA setting for several short periods of time. Whenever a newly generated solution different from the previous best is accepted, it takes a while for the pheromone values to move towards the extreme values. This results in the short downward dents visible in the plots. During these times, $P^{(t)}$ is smaller than in the (1+1) EA setting, but these dents end quickly and the MMAS returns to the (1+1) EA-like behavior.

The chart for $\rho = 0.01$ differs from the other three in the respect that only after $t = 2141$ iterations 95% of the pheromone values reach the extreme values. Also, $P^{(t)}$ stays below $1/e$ most of the time. Still, the value of $\overline{mm} \approx 982$ combined with a variance of $\overline{\nu} \approx 2$ indicate that during the central part of the run most pheromone values are either at their extreme values or at least very close to them.

Further experiments for smaller values of ρ show that the effects observed for $\rho = 0.01$ amplify. The variance $\overline{\nu}$ stays almost constant ($\overline{\nu} \approx 3.8$ for $\rho = 0.001$) indicating that most values of $p^{(t)}$ are close to the extreme values. The behavior of the MMAS remains highly (1+1) EA–like, only the performance drops due to the additional time needed for the pheromone values to reach the extreme values again after an update.

For reasons of space, we are not able to discuss typical runs for the test function LeadingOnes. However, the data displayed in Table 3 suffices to see that the optimization behavior of the MMAS for LeadingOnes

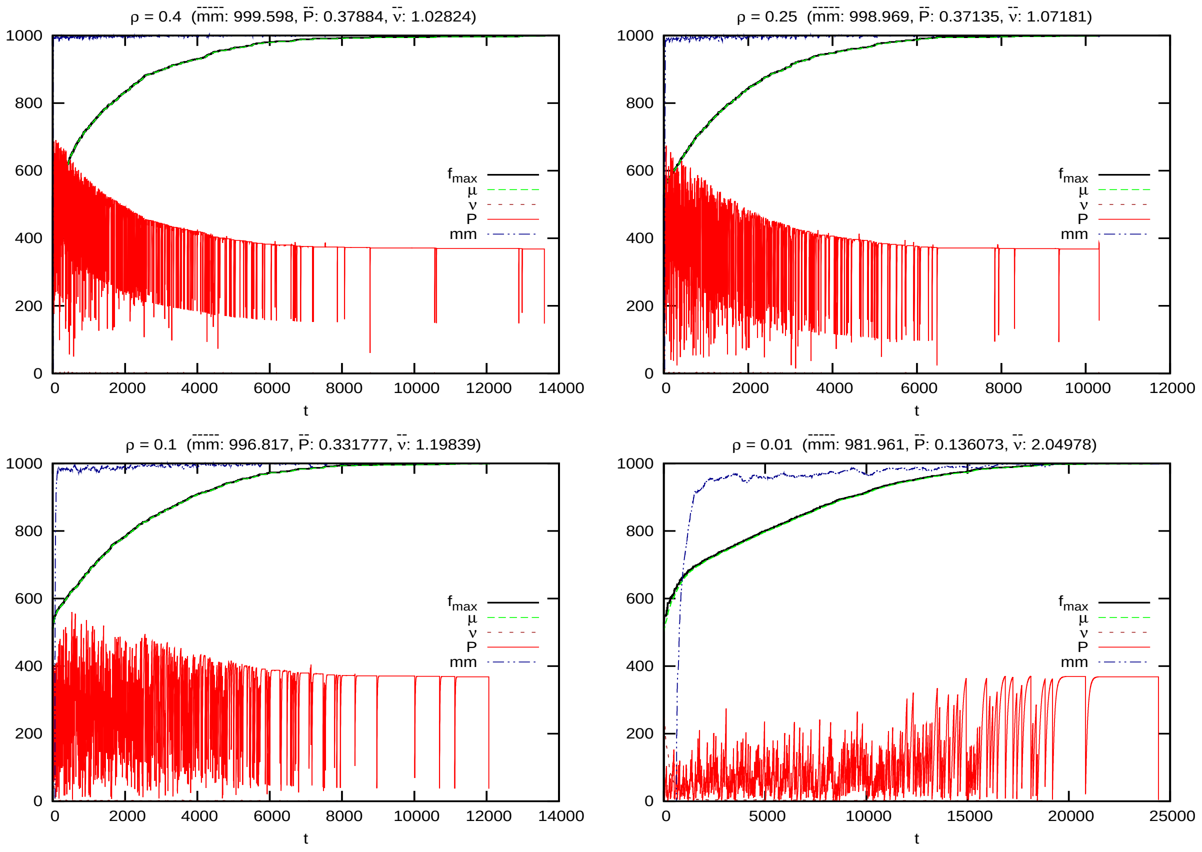

Figure 4: Four typical runs of the MMAS on OneMax for $\rho = 0.4$ (top left), 0.25 (top right), 0.1 (bottom left), and 0.01 (bottom right).

is very similar to that of the (1+1) EA, both in terms of run-times and, more declaratively, in that we have many pheromone values at or close to the extreme values of $1/n$ and $1 - 1/n$, as witnessed by $\overline{mm}$ and $\overline{\nu}$. This fact was already observed in [9] and used to prove a lower bound on the optimization time of the MMAS. Finally, also for non-leading bits there is a strong drift of the pheromone values towards $1/n$ and $1-1/n$. This observation strengthens the resemblance between the MMAS and the (1+1) EA even more.

6 Conclusion

We analyzed the two existing single ant ACO approaches for three types of fitness functions. Previous research shows that, at least for certain choices of the evaporation factor ρ, both can optimize the functions OneMax and LeadingOnes with optimization times of similar order of magnitude as the (1+1) EA.

By not only regarding the resulting optimization times, but by also monitoring well-chosen theory-guided indicators during the runs of the ACO systems, we showed that whenever the optimization time was reasonable, indeed the whole optimization behavior strongly resembles that of the (1+1) EA. Our experimental investigation also complements existing rigorous mathematical analyses in that it produces actual numbers and not only orders of magnitude. Our experiments indicate that, if existent, the advantage of single ant ACO systems over classical and technically much simpler approaches has to be shown on more advanced or non-pseudo-boolean optimization problems.

References

[1] The data is available at `http://www.mpi-inf.mpg.de/publications/index.html`.

[2] B. Doerr and D. Johannsen. Refined runtime analysis of a basic ant colony optimization algorithm. In *Proc. of the CEC 2007*, pages 501–507. IEEE Press, 2007.

[3] B. Doerr, F. Neumann, D. Sudholt, and C. Witt. On the runtime analysis of the 1-ANT ACO algorithm. In *Proc. of GECCO '07*, pages 33–40. ACM, 2007.

[4] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: An autocatalytic optimizing process. Technical Report 91-016 Revised, Politecnico di Milano, 1991.

[5] M. Dorigo and T. Stützle. *Ant Colony Optimization.* MIT Press, 2004.

[6] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *TCS*, 276:51–81, 2002.

[7] W. J. Gutjahr. First steps to the runtime complexity analysis of ant colony optimization. *Comput. Oper. Res.*, 35:2711–2727, 2008.

[8] W. J. Gutjahr and G. Sebastiani. Runtime analysis of ant colony optimization. Technical report, Mathematics department, Sapienza Univ. of Rome, 2007.

[9] F. Neumann, D. Sudholt, and C. Witt. Comparing variants of MMAS ACO algorithms on pseudo-boolean functions. In *SLS 2007*, volume 4638 of *LNCS*, pages 61–75, 2007.

[10] F. Neumann, D. Sudholt, and C. Witt. Rigorous analyses for the combination of ant colony optimization and local search. In *Proc. of ANTS 2008*, 2008. To appear.

[11] F. Neumann and C. Witt. Runtime analysis of a simple ant colony optimization algorithm. In *Proc. of ISAAC '06*, volume 4288 of *LNCS*, pages 618–627, 2006.

[12] T. Stützle and H. Hoos. MAX–MIN ant system. In *Journal of Future Generation Computer Systems*, pages 889–914, 2000.