MMAS Versus Population-Based EA on a Family of Dynamic Fitness Functions

Lissovoi, Andrei; Witt, Carsten

Published in:
Algorithmica

Link to article, DOI:
10.1007/s00453-015-9975-z

Publication date:
2015

Document Version
Peer reviewed version

Citation (APA):
Lissovoi, A., & Witt, C. (2015). MMAS Versus Population-Based EA on a Family of Dynamic Fitness Functions. *Algorithmica*, *75*(3), 554-576. https://doi.org/10.1007/s00453-015-9975-z

Algorithmica manuscript No.
(will be inserted by the editor)

MMAS versus Population-Based EA on a Family of Dynamic Fitness Functions

Andrei Lissovoi · Carsten Witt

Received: 19 October 2014

Abstract We study the behavior of a population-based EA and the Max-Min Ant System (MMAS) on a family of deterministically-changing fitness functions, where, in order to find the global optimum, the algorithms have to find specific local optima within each of a series of phases. In particular, we prove that a (2+1) EA with genotype diversity is able to find the global optimum of the MAZE function, previously considered by Kötzing and Molter (PPSN 2012, 113–122), in polynomial time. This is then generalized to a hierarchy result stating that for every μ, a (μ+1) EA with genotype diversity is able to track a MAZE function extended over a finite alphabet of μ symbols, whereas population size $\mu - 1$ is not sufficient. Furthermore, we show that MMAS does not require additional modifications to track the optimum of the finite-alphabet MAZE functions, and, using a novel drift statement to simplify the analysis, reduce the required phase length of the MAZE function.

Keywords Evolutionary Algorithms; Ant Colony Optimization; Dynamic Problems; Populations; Runtime Analysis

1 Introduction

Evolutionary algorithms (EAs) are a class of nature-inspired algorithms that can be applied to solve a wide variety of optimization problems. Runtime analysis of nature-inspired algorithms has advanced considerably in recent years [2, 12], though most focus on static optimization problems, where the objective is simply to find the global optimum within the least amount of steps. Many real-world optimization problems are dynamic in nature, meaning that the optimal solution to a given problem may change as the problem conditions change over time, and the algorithms therefore

A preliminary version of this work previously appeared in the Proceedings of the 2014 Conference on Genetic and Evolutionary Computation [11]; this version extends the presented proofs, in particular in Section 5.

DTU Compute, Technical University of Denmark, Kongens Lyngby, Denmark.

need to be able to not only find the optimum at some point of time, but also to *track* the optimal solution over time as the problem changes.

Application of EAs to Dynamic Optimization Problems is the subject of study in the Evolutionary Dynamic Optimization field, which in recent years has attracted much activity. Many applications of evolutionary algorithms on dynamic problems are considered in literature [1, 13], and there are runtime analyses building on theoretical studies of evolutionary algorithms for dynamic problems [4, 7, 15]. The utility of a population for tracking problems was studied in evolutionary computation by Jansen and Schellbach [7], while different mechanisms for ensuring population diversity have been considered by Oliveto and Zarges [14]. In particular, a mechanism called *genotype diversity* was proved to be inefficient on a particular dynamic problem. Recently, Jansen and Zarges [8] compared evolutionary algorithms and artificial immune systems on a bi-stable dynamic optimization problem.

In [9], Kötzing and Molter introduced a dynamic pseudo-boolean function called Maze, the optimum of which slowly moves from the all-ones to the all-zeroes bit string in n phases where the optimum of a single bit oscillates between the two values. The paper shows that while the Max-Min Ant System is able to track the changes occurring in this fitness function and find the optimum all-zeroes string within polynomial time, a (1+1) EA loses track of the optimum and requires an exponential amount of time to find the all-zeroes optimum with high probability.

In this paper, we consider the impact of introducing a population and a simple diversity mechanism to the (1+1) EA, showing that a (2+1) EA with genotype diversity is able to track the optimum of the Maze function. We then generalize the Maze to a function over a finite alphabet, and prove a hierarchy result with respect to the population size. More precisely, for any μ, there is a variant of the Maze such that a (μ+1) EA with genotype diversity will with high probability succeed in tracking the optimum, whereas population size $\mu - 1$ will with high probability make it lose track of the optimum. Finally, we return to consider the performance of the MMAS* Max-Min Ant System Ant Colony Optimization (ACO) algorithm, and conclude that MMAS*, due to its pheromone memory model, is always able to successfully track the optimum of the finite-alphabet version of Maze, even with shorter phase lengths than considered in [9]. Our proofs are based on mixing time arguments, fitness levels with tail bounds, and a new variant of a variable drift theorem, which allows for a tail bound on the probability distribution of the pheromone value in MMAS*.

The paper is structured as follows. Section 2 defines the dynamic fitness function Maze, and the (μ+1) EA with genotype diversity and MMAS* algorithms generalized to larger alphabets. Section 3 proves the positive result for the simple (2+1) EA w. r. t. the classical Maze function on bit strings. The hierarchy result for larger alphabets is proven in Section 4. Finally, Section 5 shows that MMAS* is efficient in tracking the optimum for every polynomial alphabet size. We finish with some conclusions.

2 Preliminaries

The Maze dynamic fitness function defined in [9] consists of $n + 1$ phases of $t_0 = kn^3 \log n$ iterations each, where k is a sufficiently large constant. During the first

Algorithm 1 (μ+1) EA with genotype diversity for a finite alphabet $\Sigma = \{0, 1, \ldots, r\}$.

```
Initialize $x^* = \{x_1, \ldots, x_\mu\}$, st. $x_i \neq x_j$ for all $i \neq j$.
for $t \leftarrow 1, 2, \ldots$ do
    Select $x_a$ from $x^*$ uniformly at random.
    Let $x_t = \mathrm{mut}_r(x_a)$.
    if $x_t \notin x^*$ then
        $G \leftarrow x^* \cup \{x_t\}$
        $x_{\min} \leftarrow \arg\min_{x \in G} f(x, t)$, chosen uniformly at random
        $x^* \leftarrow G \setminus \{x_{\min}\}$
```

phase, which we will for convenience refer to as phase 0, Maze is equivalent to OneMax. In each subsequent phase i, the function assigns fitness values $n + 2$ and $n + 1$ to bit strings $0^{i-1}01^{n-i}$ and $0^{i-1}11^{n-i}$, oscillating between assigning the higher fitness value to these individuals in a 0-0-1 pattern, with 0^i1^{n-i} being favored, i.e. having the higher fitness value, every two iterations out of three.

The version shown below has been extended to assign fitness values to n-character strings over a finite alphabet; for $r = 1$, i. e., bit strings, it is exactly equivalent to the original Maze. In this context, OneMax counts the number of literal 1 characters in the string, and the sets OPT_p and ALT_p generalize the 0-0-1 oscillation pattern.

$$
\begin{aligned}
\textsc{Maze}(x, t) &= \begin{cases} n + 2 & \text{if } t > t_0 \ \wedge \ x \in \mathrm{OPT}(t) \\ n + 1 & \text{if } t > t_0 \ \wedge \ x \in \mathrm{ALT}(t) \\ \textsc{OneMax}(x) & \text{otherwise} \end{cases} \\
\mathrm{OPT}(t) &= \begin{cases} \mathrm{OPT}_{\lfloor t/t_0 \rfloor} & \text{if } t \neq 0 \mod 3 \\ \mathrm{ALT}_{\lfloor t/t_0 \rfloor} & \text{otherwise} \end{cases} \\
\mathrm{ALT}(t) &= \begin{cases} \mathrm{ALT}_{\lfloor t/t_0 \rfloor} & \text{if } t \neq 0 \mod 3 \\ \mathrm{OPT}_{\lfloor t/t_0 \rfloor} & \text{otherwise} \end{cases} \\
\mathrm{ALL}_p &= \left\{ 0^{p-1} x 1^{n-p} \mid x \in \{0, 1, \ldots, r\} \right\} \\
\mathrm{OPT}_p &= \{0^p 1^{n-p}\} \\
\mathrm{ALT}_p &= \mathrm{ALL}_p \setminus \mathrm{OPT}_p
\end{aligned}
$$

In this paper, we will examine how a (2 + 1) Evolutionary Algorithm (EA) performs on the original Maze function, and how a (μ+1) EA performs on our finite-alphabet version. The (μ+1) EA with genotype diversity [14, 16] is shown as Algorithm 1. Definition 1 extends the mutation operator mut_r to support a finite alphabet as in [3, 5]; for $r = 1$, it is equivalent to the standard mutation operator of the (1+1) EA.

Several lemmas throughout this paper state that "a specific event occurs with high probability." Definition 2 provides a more formal definition of this concept.

Definition 1 Let $\Sigma = \{0, 1, \ldots, r\}$ be a finite alphabet.

The mutation operator mut_r creates an image $y \in \Sigma^n$ from $x \in \Sigma^n$ by independently replacing each character x_i of x ($1 \leq i \leq n$) with probability $1/n$ with a symbol drawn uniformly at random from $\Sigma \setminus \{x_i\}$.

Definition 2 An event E is said to occur with high probability if $\Pr(E) = 1 - O(n^{-c})$ for any constant $c > 0$.

Algorithm 2 The MMAS* algorithm on a directed multigraph $G = (V, E)$, with pheromone bounds $\tau_{\min}$ and $\tau_{\max}$, and evaporation rate ρ, where $v_0, v_n \in V$ are the start and destination vertices respectively, and $\deg^+(v)$ is the outdegree of a vertex.

```
Initialize $\tau_a \leftarrow 1/\deg^+(v)$ for all $a = (v, v') \in E$
for $t \leftarrow 1, 2, \ldots$ do
    Let $x_t$ be an empty path.
    $p \leftarrow v_0,\ S \leftarrow \{(p, v') \in E \mid v' \notin x_t\}$
    while $|S| > 0$ and $p \neq v_n$ do
        Select edge $e = (p, h')$ from $S$ with probability:
            $p_e = \tau_e / \sum_{s \in S} \tau_s$
        Append $e$ to $x_t$
        $p \leftarrow h',\ S \leftarrow \{(p, v') \in E \mid v' \notin x_t\}$
    if $t = 1$ or $f(x_t, t) > f(x^*, t)$ then
        $x^* \leftarrow x_t$
    for each $e \in E$ do
        $\tau_e \leftarrow \begin{cases} \min(\tau_{\max}, (1-\rho)\tau_e + \rho) & \text{if } e \in x^* \\ \max(\tau_{\min}, (1-\rho)\tau_e) & \text{otherwise} \end{cases}$
```

In Section 5, we will consider how the Max-Min Ant System [17] algorithm MMAS*, shown as Algorithm 2, is able to track the optimum of the finite-alphabet Maze function.

To use a path-constructing algorithm with a fitness function that assigns values to n-character strings, we use the construction graph shown in Figure 1: every n-character string $x \in \Sigma^n$ corresponds to the path from v_0 to v_n consisting of the edges $e_{i,c}$ for which $x_i = c$, and every v_0-v_n path corresponds to some $x \in \Sigma^n$ in this fashion. We use the standard choice for $\tau_{\max} = 1 - 1/n$, and set $\tau_{\min} = 1/(rn)$ to accommodate an $(r + 1)$-character alphabet of the extended Maze function. Notably, for $r > 1$, the sum of the pheromone values on edges leaving a vertex is no longer always equal to 1, and, when considering probabilities of selecting a particular edge, we have to use the bounds presented in Lemma 3, similar to [19, Lemma 1] and [18, Lemma 15].

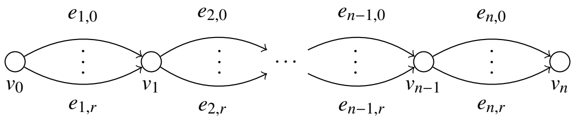

Fig. 1 Construction graph used for MMAS* on the finite-alphabet Maze function. There are $r + 1$ edges between each pair of vertices (v_{i-1}, v_i).

Lemma 3 *The sum of the pheromone values on edges leaving any specific vertex v, τ_{sum}, can be bounded as:*

$$1 \leq \tau_{sum} \leq 1 + (deg^+(v) - 1)\tau_{min} = 1 + 1/n,$$

where $deg^+(v)$ is the out-degree of vertex v.

Proof Recall that $\deg^+(v) = r + 1$ and $\tau_{\min} = 1/(rn)$. We prove these bounds by induction, noting that both hold at initialization, where $\tau_{\text{sum}} = 1$.

If, prior to a pheromone update, $\tau_{\text{sum}} \geq 1$, and no pheromone values are affected by the $\tau_{\max}$ border, $\tau_{\text{sum}}(1-\rho)+\rho \geq 1$ after the update; while if there are pheromone values capped at $\tau_{\max}$, we note that even if all the other pheromone values are $\tau_{\min}$, $\tau_{\max} + r \cdot \tau_{\min} = 1$, proving the first inequality.

If, prior to a pheromone update, $\tau_{\text{sum}} \leq 1 + 1/n$, $\tau_{\text{sum}}(1-\rho)+\rho \geq \tau_{\text{sum}}$ can only occur as a consequence of pheromone values being affected by the lower pheromone border (as $\tau_{\text{sum}} \geq 1$), i. e., those for which $\tau(1-\rho) \leq \tau_{\min}$, increasing τ_{sum} by at most $\rho\tau_{\min}$ for each such value. We note that there can be at most r such values, as the reinforced pheromone value cannot drop below $\tau_{\min}$, thus the sum of pheromone values after the update is at most $\tau_{\text{sum}}(1-\rho)+\rho+r\cdot\rho\tau_{\min} = \tau_{\text{sum}}(1-\rho)+\rho+\rho/n \leq 1 + 1/n$, proving the second inequality. □

Note that both algorithms re-evaluate the fitness function when updating the population or the best-so-far solution. Similarly to [8], the considered clock t is external to the Maze function, making it possible to evaluate many solutions in one clock tick of the Maze; this corresponds to being able to having hardware available to evaluate many solutions in parallel, while the problem changes occur at fixed intervals regardless of the number of parallel evaluations.

3 (2+1) EA on Maze

The (1+1) EA will with high probability require an exponential amount of time to find the 0^n optimum on the Maze function, because there is at least a constant probability of ending each phase $p > 0$ with $x^* \neq 0^p 1^{n-p}$, which lets the (1+1) EA revert to optimizing OneMax, destroying the 0-prefix constructed so far, and eventually requiring a large mutation to recover [9]. In this section, we will show that a (2+1) EA with genotype diversity avoids this problem by being able to store both of the oscillating individuals in each phase, thereby ensuring that $0^p 1^{n-p}$ is in the population at the start of the next phase. This ensures that at the end of the last phase, the 0^n individual is in the population with high probability.

Theorem 4 *The* $(2+1)$ *EA with genotype diversity will with high probability have the* 0^n *optimum in the population at the end of the last phase of the* Maze.

To show this, we will prove that the $1^n = \text{OPT}_0$ individual is found with high probability during the initial OneMax phase, and the EA is then able to follow the oscillation process: if OPT_{p-1} is in the population at the beginning of phase p, OPT_p will be in the population at the end of that phase – meaning that at the end of final phase, $\text{OPT}_n = 0^n$ will be in the population.

Lemma 5 *The* $(2+1)$ *EA will discover the* 1^n *individual within* $O(n \log n)$ *iterations with high probability.*

Proof The fitness level method can be applied: partition the possible populations into levels based on the maximum fitness value of any individual within the population. The probability of an iteration leaving the level where the highest fitness value is $n - i$ is at least $p_i \geq (1/2) \cdot (1/n) \cdot (1-1/n)^{n-1} \geq i/(2ne)$: i. e., that of selecting the best-valued individual in the population as the ancestor, and flipping a single zero bit. Even if the EA starts at the lowest of $n + 1$ fitness levels, it has to leave at most n levels in order to reach the level where 1^n is in the population; in expectation, this takes $\mathrm{E}(T) \leq \sum_{i=1}^{n} \frac{1}{p_i} = 2en \log n$ iterations.

The high probability result is obtained by applying the tail bounds on fitness levels derived in [20, Theorem 2]: using $s = 50n^2 > 4\pi^2 e^2 n^2/6 \geq \sum_{i=1}^{n} \frac{1}{{p_i}^2}$, $h = 1/(2en) \leq \min p_i$, the probability that 1^n is found within $\mathrm{E}(T) + \delta$ iterations is at least $1 - e^{-\frac{\delta}{4} \cdot \min\{\delta/s, h\}}$. Setting $\delta = 50cn \log n$, where $c > 0$, and observing that $h < \delta/s$ for sufficiently large n, yields a probability of finding 1^n within $O(n \log n)$ iterations of at least

$$1 - e^{-\frac{50cn \log n}{4} \cdot \min\left\{\frac{50cn \log n}{50n^2}, \frac{1}{2en}\right\}} \geq 1 - e^{-\frac{50c \log n}{8e}} > 1 - n^{-2c}. \quad \square$$

Lemma 6 *If OPT_{p-1} is in the population at the beginning of phase p, OPT_p will with high probability be in the population when phase p ends.*

Proof As $\mathrm{OPT}_{p-1} \in \mathrm{ALL}_p$, it has a fitness value of at least $n + 1$ during phase p, and therefore cannot be removed from the population. The OPT_p individual can be constructed by selecting OPT_{p-1} as the ancestor and flipping a single bit, which occurs in each iteration with probability at least $1/(2en)$. The probability of this occurring within $n^3 \log n$ iterations is at least:

$$1 - (1 - 1/(2en))^{n^3 \log n} \geq 1 - e^{-\frac{1}{2e} n^2 \log n} = 1 - n^{-\Omega(n^2)}.$$

As OPT_p has a fitness value of at least $n + 1$ during phase p, when constructed, it will replace a OneMax-valued individual in the population, and cannot be removed from the population during phase p. $\square$

Combined, Lemma 5 and Lemma 6 prove Theorem 4.

Proof (of Theorem 4) By applying a union bound on the probabilities of failure during each of Maze's $n + 1$ phases, we can conclude that the (2+1) EA finishes phase n with 0^n in the population with high probability. $\square$

4 (μ+1) EA and the finite-alphabet Maze

While a (2+1) EA with genotype diversity is able to track the optimum of the original Maze, it is interesting to consider whether there exist Maze-like functions for which a larger population is required. In this section, we use the Maze function extended over a finite alphabet $\Sigma = \{0, 1, \ldots, r\}$, and consider a ($\mu$+1) EA with genotype diversity, where $r \in O(n)$ and $\mu < n/2$. We will build toward two results: a population of $\mu \leq r$ is insufficient to track the optimum, while $\mu > r$ is sufficient and enables the (μ+1) EA to find 0^n in polynomial time. These results are formalized in Theorems 7 and 8.

Theorem 7 *If $r \geq \mu$, $r \in O(n)$, and $\mu \leq n/(2 + \epsilon)$, where $\epsilon > 0$ is an arbitrarily small constant, (μ+1) EA with genotype diversity will with high probability not find the 0^n optimum on* Maze *within a polynomial number of iterations.*

Theorem 8 *If $\mu, r \in O(n)$ and $r < \mu$, (μ+1) EA with genotype diversity will with high probability finish the last phase of the* Maze *with the* 0^n *optimum in the population.*

As before, we need to verify that 1^n is found during the initial OneMax phase; this is done in Lemma 9. Then, Lemma 10 shows that if an iteration begins with one of ALL_p individuals (i. e., those with a non-OneMax value during that phase) in the population, the population will with high probability be saturated with ALL_p individuals before the phase is over. These two lemmas are used in the proofs of both Theorems 7 and 8.

Lemmas 11 and 12 are used for Theorem 7. The former shows that once a population of $\mu \le r$ individuals is filled with ALL_p individuals, the probability of OPT_p being in the population is at most a constant after a small number of additional iterations; while the latter states that if a phase p begins with no ALL_p individuals in the population (i. e., OPT_{p-1} was not in the population when the phase $p-1$ ended), the (μ+1) EA loses track of the optimum and reverts to optimizing OneMax with at least constant probability. This proof strategy is inspired by the lower bound for the (1+1) EA on the original Maze [9].

Lemma 9 *The (μ+1) EA will discover the* 1^n *individual within* $O(\mu r n \log n) = O(n^3 \log n)$ *iterations with high probability.*

The method used to prove Lemma 5 can be applied to prove Lemma 9 as well: we require that the best-fitness individual in the population is chosen as the ancestor, and the mutation operator changes a single non-1 character into a 1, resulting in an individual with a higher OneMax-value than any previously in the population.

Lemma 10 *During phase p, once an individual from ALL_p is in the population, the population will contain* $\min(\mu, r+1)$ *individuals from ALL_p within* $O(\mu r n \log n)$ *iterations with high probability.*

Proof The general form of the fitness level method can be applied by partitioning the μ-individual populations into levels by the number of ALL_p individuals they contain, from 0 to $\min(\mu, r+1)$, observing that the number of ALL_p individuals in the population cannot be reduced during phase p. As the phase starts with a population containing at least one ALL_p individual, there are at most $\min(\mu, r+1) - 1$ "population levels" that the process may need to pass through before the population is saturated with ALL_p individuals; the time for this to happen can be bounded as a sum of geometrically distributed waiting times to leave each "level".

If $i < \min(\mu, r+1)$ is the number of ALL_p individuals in the population, the probability of a single iteration creating a new ALL_p individual (and hence moving to a higher level) by selecting one of the i as the ancestor and performing a one-character mutation is $p_i \ge \frac{i}{\mu}\frac{r+1-i}{r}\frac{1}{en}$. Let T be the number of iterations before the population is saturated with ALL_p individuals, and $m = \min(\mu, r+1) - 1$ be the number of ALL_p individuals that need to be added to the population to achieve saturation, thus the expectation $\text{E}(T)$ is at most:

$$\sum_{i=1}^{m} \frac{\mu}{i} \frac{r}{r+1-i} en = r\mu en \sum_{i=1}^{m} \frac{1}{i(r+1-i)} = O(r\mu n).$$

Applying the tail bounds from [20, Theorem 2]: using $s = 13\mu^2 r^2 n^2 > \sum_{i=1}^{m} \frac{1}{{p_i}^2}$, $h = 1/(\mu r e n) < \min p_i$, the probability that m ALL_p individuals are added to the population within $\mathrm{E}(T) + \delta$ iterations is at least $1 - e^{-\frac{\delta}{4} \cdot \min\{\delta/s, h\}}$. Setting $\delta = 13 c \mu r n \log n$, where $c > 0$, and observing that $h < \delta/s$ for sufficiently large n, yields a probability of reaching the final level within $O(\mu r n \log n)$ iterations of at least

$$1 - e^{-\frac{13 c r \mu n \log n}{4} \cdot \min\left\{\frac{13 c r \mu n \log n}{13 \mu^2 r^2 n^2}, \frac{1}{e r \mu n}\right\}} \geq 1 - e^{-\frac{13 c \log n}{4e}} > 1 - n^{-c}.$$

Thus, if an ALL_p individual exists in the population during phase p, the population will be saturated with ALL_p individuals in $O(\mu r n \log n)$ iterations with high probability. □

The following lemmas consider the situation for $\mu \leq r$, i. e., a population size too small to contain every individual in ALL_p. In this case, OPT_p can be removed from the population after being discovered. Interestingly, this happens with constant probability regardless of μ.

Lemma 11 *If, during phase p, the population consists of μ individuals from ALL_p, and $r \geq \mu$, then after $\Omega(rn)$ iterations, the probability that OPT_p is in the population is at most a constant smaller than 1.*

Proof OPT_p can be replaced by an ALT_p individual that was not in the population during an iteration that favors ALT_p individuals over OPT_p. The probability p_L of OPT_p being replaced by one of $r + 1 - \mu$ ALT_p individuals not yet in the population can then be bounded:

$$p_L = \frac{r + 1 - \mu}{r} \frac{1}{n} \left(1 - \frac{1}{n}\right)^{n-1},$$
$$\frac{1}{rne} \leq p_L \leq \frac{1}{n}.$$

The OPT_p individual can be added to the population during any iteration favoring it over the ALT_p individuals by a single-character mutation of any individual in ALT_p. The probability p_W of OPT_p being rediscovered during an iteration favoring it can be bounded:

$$p_W = \frac{1}{r} \frac{1}{n} \left(1 - \frac{1}{n}\right)^{n-1},$$
$$\frac{1}{rne} \leq p_W \leq \frac{1}{rn}.$$

Consider the probability of the OPT_p individual being in the population after three additional iterations: it is greatest when the first of these three iterations favors ALT_p individuals, and the subsequent two favor OPT_p. Let p_{W3} be the probability that OPT_p is constructed during the two iterations favoring it (being added to the population if it was not already there):

$$p_{W3} = p_W + (1 - p_W) \cdot p_W,$$
$$\frac{1}{rne} \leq p_W \leq p_{W3} \leq 2 p_W \leq \frac{2}{rn},$$

and p_{L3} be the probability that OPT_p is replaced by an ALT_p individual and then not constructed in the two following iterations:

$$p_{L3} = p_L \cdot (1 - p_{W3}),$$
$$\frac{1}{2rne} \leq \frac{1}{rne}\frac{rn-2}{rn} \leq p_{L3} \leq \frac{1}{n}\left(1 - \frac{1}{rne}\right) < \frac{1}{n},$$

where the lower bound holds for $n \geq 4$.

This behavior can be modeled in a two-state Markov chain, where the states represent having or not having OPT_p in the population, with transition probabilities p_{W3} and p_{L3}; each step of this Markov chain thus corresponds to three iterations of the (μ+1) EA. Let π_H be the steady-state probability of the OPT_p individual being in the population:

$$\pi_H \cdot p_{L3} = (1 - \pi_H) \cdot p_{W3}$$
$$\frac{1 - \pi_H}{\pi_H} = \frac{p_{L3}}{p_{W3}} \geq \frac{1}{4e}$$
$$\pi_H \leq \frac{1}{1 + \frac{1}{4e}} < 0.916.$$

Assume, as the worst case, that the OPT_p individual is in the population when it becomes saturated with ALL_p individuals. Markov chain mixing time, $t(\epsilon)$, can then be used to find the number of steps required to reduce the total variation distance to the steady state π_H to at most some ϵ (and hence the number of iterations required to reduce the probability of OPT_p being in the population to at most some constant smaller than 1), and can be upper-bounded by coupling time as in [18, Corollary 4]:

$$t(\epsilon) \leq \min\left\{t : \max_{x,y\in\Omega} \Pr(T_{xy} > t) \leq \epsilon\right\},$$

where $T_{xy} = \min\{t : X_t = Y_t \mid X_0 = x, Y_0 = y\}$ is the coupling time, i. e., the earliest time at which the two equivalent Markov chains X_t and Y_t, initialized in different states x, y, are in the same state.

With only two states (and symmetry), the coupling time T_{xy} is greatest when the chains begin in different states. The probability that the chains remain in different states for at least t steps is then:

$$\max_{x,y\in\Omega} \Pr(T_{xy} > t) = \left(p_{L3}p_{W3} + (1 - p_{L3})(1 - p_{W3})\right)^t.$$

To get an upper bound on $t(\epsilon)$, an upper bound on the expression in parentheses is needed. Inserting the appropriate bounds on p_{W3} and p_{L3} yields:

$$2p_{W3}p_{L3} + 1 - p_{L3} - p_{W3} \leq \frac{4}{rn^2} + 1 - \frac{3}{2rne}$$
$$\leq 1 - \frac{1}{rne}\left(1.5 - \frac{4e}{n}\right) < 1 - \frac{1}{rne} \text{ (for } n \geq 22\text{)},$$
$$t(\epsilon) \leq \min\left\{t : \left(1 - \frac{1}{rne}\right)^t \leq \epsilon\right\}.$$

After, e. g., $t(0.01) < 4.61rne$ steps of the Markov chain, the probability that coupling has not occurred is at most 0.01. As coupling time is an upper bound for mixing time, and each Markov chain step corresponds to three EA iterations, this means that after at most $37.60rn$ iterations, the probability that OPT_p is in the population is at most $\pi_H + 0.01$, and so the probability that OPT_p is not in the population is at least 0.083.

Therefore, $\Omega(rn)$ iterations after the population is saturated with ALL_p individuals, the probability that OPT_p is in the population is at most a constant. □

Lemma 11 therefore implies that there is at least a constant probability of phase $p+1$ beginning with only ALT_p individuals in the population when $r \geq \mu$. The following lemma considers the consequences of this – as OPT_p is the only individual in both ALL_p and ALL_{p+1}, this means that phase $p+1$ begins without any ALL_{p+1} individuals in the population, leaving the EA with only OneMax-valued individuals at the beginning of phase $p+1$.

Lemma 12 *If at the beginning of phase $p \geq n/2 + 4$, the population contains only individuals from ALT_{p-1}, with at least constant probability, the population at the end of phase p will consist only of* 1^n *and one-character mutations of* 1^n.

Proof The ALT_{p-1} individuals initially in the population are of the form $0^{p-2}x1^{n-p+1}$, where $x \in \Sigma \setminus \{0\}$, and have OneMax-values of at least $n - p + 1$. If any ALL_p individual (which are all of the $0^{p-1}y1^{n-p}$ form, where $y \in \Sigma$) is constructed, it will be accepted, and the optimization process will resume according to Lemma 10. If no such individual is constructed, individuals with higher OneMax-values will continue being accepted into the population, eventually leading to the discovery of 1^n as in Lemma 9.

Until an ALL_p individual has been constructed, the probability of constructing one can be upper-bounded by the probability of a mutation changing a character at a specific position (the x in the description of ALT_{p-1} above) to a 0, i. e., $1/(rn)$.

Let a *value-improving mutation* be a mutation that produces an individual with a strictly higher OneMax-value than the worst individual currently in the population. Note that even if a value-improving mutation occurs, the resulting individual might already be one of the $\mu - 1$ other individuals in the population, in which case the population is not modified. We will now show that the probability p_I of such a mutation occurring while the minimum OneMax-value of any individual in the population is below $n - p + 4$ is at least:

$$p_I \geq \frac{n}{2}\frac{1}{n}\frac{1}{r}\left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{2er}.$$

Let $n - p + 1 \leq v \leq n - p + 3$ be the minimum OneMax-value of an individual in the population. If the ancestor selected during an iteration has a OneMax-value of at least $v + 2$, then *any* one-character mutation will produce an individual of sufficient value to be accepted; there are $n \geq n/2$ such mutations. If the selected ancestor has a value of at most $v + 1$, it contains at least $n - (n - p + 4) \geq n/2$ non-1 characters, and hence at least $n/2$ one-character mutations can produce an individual with a strictly higher OneMax-value than such an ancestor. Thus, regardless of the ancestor selection, there are at least $n/2$ possible one-character mutations leading to an acceptable fitness value.

No more than a constant fraction of those value-improving mutations can fail to be accepted due to already being in the population (as $\mu \leq n/(2+\epsilon)$). This means that the probability of a value-improving mutation occurring and being accepted during a single iteration is at least $\Omega(p_I) = \Omega(1/r)$.

Consider the probability of the minimum OneMax-value of the population rising to at least $n - p + 4$ without any ALL_p individual being constructed; this requires that at most 3μ value-improving mutations to be accepted (as a value-improving mutation can only introduce an individual with the same fitness value into the population at most μ times; of course, it might also require as few as μ mutations). This occurs with at least constant probability: let A be the event that an ALL_p individual is constructed, and V be the the event that a value-improving mutation is accepted, then:

$$\Pr(A \mid A \vee V) \leq \frac{1/(rn)}{1/(rn) + \Omega(1/r)} = O(n^{-1}),$$
$$\Pr(3\mu\ V\text{s without } A) \geq (1 - \Pr(A \mid A \vee V))^{3\mu} = (1 - O(n^{-1}))^{O(n)} = \Omega(1).$$

Once the minimum OneMax-value of the population is raised to $n - p + 4$, constructing an ALL_p individual requires at least three characters to be changed into a 0 simultaneously, which occurs with probability at most $1/(rn)^3$.

Thus, with at least constant probability, if phase p begins without an ALL_p individual in the population, no ALL_p individual is constructed within the $O(\mu rn \log n)$ iterations required to find the 1^n individual with high probability (per Lemma 9). Furthermore, once the 1^n individual is in the population, one-character mutations of that individual will fill the population in $O(\mu \log n)$ iterations with high probability, making construction of an ALL_p individual require a $p - 2 = \Omega(n)$ character mutation. □

By combining these lemmas, it is now possible to prove Theorems 7 and 8.

Proof (of Theorem 7) Assume the EA successfully follows the Maze through the first $n/2 + 2$ phases. Each of the remaining $\Omega(n)$ phases has at least a constant probability of not ending with the OPT_p individual in the population, causing the next phase to begin with a population filled with ALT_p individuals.

With at least a constant probability, each one the remaining phases that begin without an ALL_p individual will cause the EA to fill the population with one-character mutations of 1^n. Once this occurs, constructing an ALL individual (for this or any future phase) will require at least $\Omega(n)$ characters to be mutated correctly in a single iteration, which occurs with probability $(rn)^{-\Omega(n)}$.

Therefore, with probability $1 - 2^{-\Omega(n)}$, the EA will fail to track the oscillation, revert to optimizing OneMax, and require an exponential number of iterations to find the final 0^n optimum. □

Proof (of Theorem 8) Per Lemma 9, the EA is able to find the 1^n individual with high probability during phase 0. For the subsequent n oscillation phases, apply Lemma 10:

if the phase begins with OPT_{p-1} (i. e., an individual from ALL_p) in the population, then the remaining individuals from ALL_p, including OPT_p will be added to the population before phase p ends with high probability. Add up the failure probabilities in these $n + 1$ phases using a union bound; with high probability, none of the phases fail, and phase n ends with $\mathrm{OPT}_n = 0^n$ in the population – so the (μ+1) EA is able to find the 0^n optimum within a polynomial number of iterations with high probability when $\mu > r$. □

5 ACO on larger alphabets

Kötzing and Molter [9] study the case where $r = 1$, i. e., the case of bit strings, and show that MMAS* with overwhelming probability will track the optimum of Maze with oscillation phase length $t_0 = kn^3 \log n$ in polynomial time. Their proof can be summarized as follows, wherein all references to lemmas and theorems refer to the paper:

1. While a bit is oscillating, taking the total over three steps of a so-called OPT-OPT-ALT oscillation (where OPT is favored in the first two steps and ALT in the last step) there is a drift of the pheromone value on the 0-edge of the bit (i. e., the edge associated with OPT) towards its maximum. With overwhelming probability, the pheromone value will reach its upper border in $O(n^3 \log n)$ steps (Lemma 2 in their paper) provided the bit keeps oscillating so long. To obtain the bound on the probability, a multiplicative drift theorem is used, which, according to [9], “wastes a lot” since the bound on the drift obtained is additive.
2. Despite the fact that the pheromone value reaches its upper border within the oscillation phase of a bit, the multiplicative drift theorem does not imply that the pheromone value stays close to the border by the end of the oscillation phase. To prove that this is unlikely, a negative drift theorem is applied (Lemma 3), implying that it takes a large polynomial amount of time until the pheromone value drops below $1 - O((\log^2 n)/n)$.
3. The transition from “bit i oscillating” to “bit $i + 1$ oscillating” is analyzed. It is shown that a string outside ALL will be best-so-far string only temporarily for at most $O(\log n)$ steps after the transition, with high probability (Lemma 3). Drift arguments towards the 0-value are applied afterwards.
4. Using an $O((n \log n)/\rho)$ bound from the literature on the time from initialization until the all-ones string with is found high probability (Lemma 5) and basically applying union bounds leads to the result (Theorem 6).

It turns out that the above analysis to a very large extent carries over to larger alphabets. Basically, one can group together all edges belonging to the non-0 entries of a character and identify the sum of their pheromone values with the pheromone on the 1-edge in the binary case. The only thing to pay attention to is the new lower bound on the pheromone values, which may increase the time to reach the upper border by a factor of r.

However, we are going to present a stronger and, as we think, more elegant analysis here. In particular, we contribute a technique that supplements the drift result from

Lemma 3 in [9] with a statement on probability distributions (also called occupation probabilities), which makes the application of the negative drift theorem (Lemma 4 in [9]) unnecessary. In addition, the stronger analysis allows us to work with shorter oscillation phase lengths, as detailed below.

We now present the tool by which the statement on occupation probabilities is obtained. The following lemma is a variable drift theorem that is not concerned with the expected first hitting time of a set of target states but with the probability of being in this set at any time t (however, results on the hitting time can be easily obtained from this probability). It is a spin-off of the variable drift theorem with tail bounds from [10] and goes back to a statement from Hajek's paper [6] that, to the best of our knowledge, has not been applied in the running time analysis of evolutionary algorithms yet. The lemma requires a bound on the moment-generating function of a potential function g that is usually derived from the one-step drift (via the function h).

Lemma 13 *Let $(X_t)_{t\geq 0}$, be a stochastic process, adapted to a filtration $(\mathcal{F}_t)_{t\geq 0}$, over some state space $S \subseteq \{0\} \cup [x_{\min}, x_{\max}]$, where $x_{\min} \geq 0$. Let $a, b \in \{0\} \cup [x_{\min}, x_{\max}]$, $b > a$. Let $h\colon [x_{\min}, x_{\max}] \to \mathbb{R}^+$ be such that $1/h$ is integrable on $[x_{\min}, x_{\max}]$ and define $g\colon \{0\} \cup [x_{\min}, x_{\max}] \to \mathbb{R}^{\geq 0}$ by $g(x) := \frac{x_{\min}}{h(x_{\min})} + \int_{x_{\min}}^{x} \frac{1}{h(y)}\,\mathrm{d}y$ for $x \geq x_{\min}$ and $g(0) := 0$.*

If there exist $\lambda > 0$, $\beta < 1$ and $D > 0$ such that

$$\begin{aligned} \mathrm{E}(e^{-\lambda(g(X_t)-g(X_{t+1}))} \cdot \mathbb{1}\{X_t > a\} \mid \mathcal{F}_t) &\leq \beta \\ \textit{and}\ \mathrm{E}(e^{-\lambda(g(a)-g(X_{t+1}))} \cdot \mathbb{1}\{X_t \leq a\} \mid \mathcal{F}_t) &\leq D, \end{aligned}$$

then

$$\Pr(X_t \geq b \mid X_0) < \beta^t \cdot e^{\lambda(g(X_0)-g(b))} + \frac{1-\beta^t}{1-\beta} D e^{\lambda(g(a)-g(b))}$$

for $t > 0$.

Proof We use ideas implicit in in the proof of Inequality 2.6 of [6], which uses the exponential method (a generalized Chernoff bound), and argue

$$\begin{aligned} \Pr(X_t \geq b \mid X_0) = \Pr(g(X_t) \geq g(b) \mid X_0) &= \Pr(e^{\lambda g(X_t)} \geq e^{\lambda g(b)} \mid X_0) \\ &\leq \mathrm{E}(e^{\lambda g(X_t) - \lambda g(b)} \mid X_0), \end{aligned}$$

where the first inequality uses that $g(x)$ is non-decreasing, the equality that $x \mapsto e^x$ is a bijection, and the last inequality is Markov's inequality. Now,

$$\begin{aligned} \mathrm{E}(e^{\lambda g(X_t)} \mid X_0) &= \mathrm{E}\left(e^{\lambda g(X_{t-1})} \cdot \mathrm{E}(e^{-\lambda(g(X_{t-1})-g(X_t))} \mid \mathcal{F}_{t-1})\right) \\ &= \mathrm{E}\left(e^{\lambda g(X_{t-1})} \cdot \mathrm{E}(e^{-\lambda(g(X_{t-1})-g(X_t))} \cdot \mathbb{1}\{X_{t-1} > a\} \mid \mathcal{F}_t)\right) \\ &\quad + \mathrm{E}\left(e^{\lambda g(X_{t-1})} \cdot \mathrm{E}(e^{-\lambda(g(a)-g(X_t))} \cdot \mathbb{1}\{X_{t-1} \leq a\} \mid \mathcal{F}_t)\right) \\ &\leq \beta \mathrm{E}\left(e^{\lambda g(X_{t-1})} \mid X_0\right) + D e^{\lambda g(a)} \end{aligned}$$

by our prerequisites; we omitted the condition on X_0 for space reasons in lines 2–4. Inductively (note that this does not assume independence of the $g(X_{t-1}) - g(X_t)$), we get

$$\mathrm{E}\left(e^{\lambda g(X_t)} \mid X_0\right) \le \beta^t e^{\lambda g(X_0)} + \sum_{r=0}^{t-1} \beta^r D e^{\lambda g(a)},$$

altogether

$$\Pr(X_t \ge b \mid X_0) \le e^{\lambda(g(X_0)-g(b))}\beta^t + \frac{1-\beta^t}{1-\beta} D e^{\lambda(g(a)-g(b))}$$

as suggested. □

The preceding lemma will be applied to show Lemma 15, whose purpose is to combine Lemmas 2 and 3 from [9] and which shows that the pheromone value after a certain point of time will come close to its border and stay close with high probability. In fact, due to the strength of the lemma, we can work with an oscillation length of $\Theta(r^2 n^2 \ln(rn))$ per character, i. e., we may (but need not) redefine $t_0 := kr^2n^2 \ln(rn)$ in the Maze function, for some constant k. By contrast, [9] needs $\Omega(n^3 \log n)$ oscillations per bit due to the application of the multiplicative drift theorem along with negative drift.

To prepare the proof of Lemma 15, we set up an appropriate potential function to define a non-negative stochastic processs $(X_t)_{t\ge 0}$, which is partially based on the distance of the pheromone value to its border. However, as in [9], it also includes an indicator random variable based on the best-so-far entry. The following lemma summarizes some crucial properties of the process. Hereinafter, we call the pheromone values of a character i, $1 \le i \le n$, *saturated* according to the best-so-far solution x^* if the edge corresponding to x^*_i has pheromone $\tau_{\max}$ and all other edges for character i have value $\tau_{\min}$.

Lemma 14 *Assume $\rho \le \frac{1}{7rn}$ and $\rho = \Omega(\frac{1}{rn})$. Consider an OPT-OPT-ALT oscillation of a single character, with the best-so-far solution from ALL, and pheromone values of the non-oscillating characters saturated according to the best-so-far solution. Let τ_t be the pheromone value on the edge corresponding to the* 0*-entry for the oscillating character after t OPT-OPT-ALT oscillations, C_t be an indicator that the best-so-far solution at the start of the oscillation is OPT, and $X_t := 1 - 1/n - \tau_t + \frac{7}{2}\rho(1 - C_t)$ be a potential function. Then, for $X_t \ge 1/n$, the following observations hold:*

1. $\Pr(X_{t+1} > X_t \mid C_t = 1) = O(X_t)$ *and* $\Pr(X_{t+1} < X_t \mid C_t = 0) = O(1 - X_t)$,
2. $(X_t - X_{t+1} \mid C_t = 1) = O(X_t\rho)$ *and* $(X_{t+1} - X_t \mid C_t = 0) = O((1 - X_t)\rho)$,
3. $\mathrm{E}(X_t - X_{t+1} \mid C_t = 1) = \Omega(X_t\rho)$ *and* $\mathrm{E}(X_t - X_{t+1} \mid C_t = 0) = \Omega((1 - X_t)\rho)$.

Proof Consider the statements conditioned on $C_t = 1$ first: i. e., those that consider oscillations that begin with OPT as the best-so-far solution. In such oscillations, the first two iterations will always reinforce τ_t, as the best-so-far solution OPT has the highest possible fitness value; while the last iteration may replace OPT with a solution from ALT if one is constructed.

We note that the potential function X_t decreases unless an ALT solution is constructed in the third iteration, and upper-bound this probability using τ_t, noting that iterations reinforcing τ_t do not increase the probability that ALT solutions are constructed:

$$\Pr(X_{t+1} > X_t \mid C_t = 1) < 1 - \tau_t = X_t + \tfrac{1}{n} \leq 2X_t,$$

as $X_t \geq 1/n$.

If an ALT solution is not constructed, τ_t is reinforced thrice during the oscillation, decreasing X_t by at most:

$$\begin{aligned}(X_t - X_{t+1} \mid C_t = 1) &\leq \tau_t(1-\rho)^3 + \rho\left(1 + (1-\rho) + (1-\rho)^2\right) - \tau_t \\ &= (1-\tau_t)\rho(3 - 3\rho + \rho^2) < 3(X_t + \tfrac{1}{n}) = 6X_t\rho,\end{aligned}$$

as $X_t \geq 1/n$.

Finally, for the expected decrease in X_t, bound the probability of constructing ALT in the third iteration using $p_a \leq (1-\tau_t/\tau_{\text{sum}})\tau_{\max}{}^{n-1} \leq (1-\tau_t/\tau_{\text{sum}})/2$, where τ_{sum} denotes the sum of pheromone values on the edges belonging to the character. The upper bounds equals the probability of selecting an oscillating edge corresponding to an ALT solution, and $n-1$ other edges with pheromone values $\tau_{\max}$ corresponding to the remaining characters in the ALT solution, using that $\tau_{\max}{}^{n-1} \leq 1/2$ for $n > 1$. Hence,

$$\begin{aligned}\mathrm{E}(\tau_{t+1} \mid C_t = 1) &= \tau_t(1-\rho)^3 + \rho(1-\rho)^2 + \rho(1-\rho) + (1-p_a)\rho, \\ \mathrm{E}(X_t - X_{t+1} \mid C_t = 1) &= \mathrm{E}(\tau_{t+1} \mid C_t) - \tau_t - p_a \cdot \left(\tfrac{7}{2}\rho\right) \\ &= \rho\left((1-\tau_t)\left(3 - 3\rho + \rho^2\right) - \tfrac{9}{2}p_a\right) \\ &= \rho\left((1-\tau_t)\left(\tfrac{3}{4} - 3\rho + \rho^2\right) - \tfrac{9}{4}\tau_t/n\right) = \Omega(\rho X_t),\end{aligned}$$

recalling that $\tau_{\text{sum}} \leq 1 + 1/n$, yielding $(1 - \tau_t/\tau_{\text{sum}}) \leq 1 - \tau_t + \tau_t/n$.

In the statements conditioned on $C_t = 0$, the best-so-far solution is in ALT. The first two iterations will evaporate τ_t unless an OPT solution is constructed; while the final iteration will evaporate τ_t if an ALT solution is constructed, or if OPT was not constructed during the first two.

We note that X_t increases unless an OPT solution is constructed during at least one of the first two iterations, and upper-bound this probability using τ_t, $\tau_{\max}{}^{n-1} \leq 1/2$ for $n > 1$, and a union bound:

$$\Pr(X_t > X_{t+1} \mid C_t = 0) < 2(\tau_t/2) = \tau_t = 1 - X_t - \tfrac{1}{n} + \tfrac{7}{2}\rho < 1 - X_t.$$

If an OPT solution is not constructed, τ_t is evaporated thrice during the oscillation, increasing X_t by at most:

$$\begin{aligned}(X_{t+1} - X_t \mid C_t = 0) &= \tau_t - \tau_t(1-\rho)^3 = \tau_t\rho(3 - 3\rho + \rho^2) \\ &< 3(1 - X_t - \tfrac{1}{n} + \tfrac{7}{2}\rho)\rho < 3(1 - X_t)\rho.\end{aligned}$$

Finally, to compute the expected decrease in X_t, we need to derive bounds for three probabilities: p_o, the probability that OPT is constructed at least once in the first two iterations; p_f, the probability that given OPT is constructed in the first two iterations, it was constructed in the first iteration; and p_a, the probability that given that OPT was constructed in the first two iterations, an ALT solution is constructed in the third iteration.

Using ${\tau_{\max}}^{n-1} \geq 1/e$, we upper-bound the probability that OPT is *not* constructed by $1 - p_o \leq (1 - \tau_t/(e\tau_{\mathrm{sum}}))(1 - \tau_t(1-\rho)/(e\tau_{\mathrm{sum}}))$, which provides a lower bound on p_o.

In most cases, OPT is more likely to be constructed in the first iteration compared to the second, as its corresponding pheromone value would decrease due to pheromone evaporation. However, if $\tau_t(1-\rho)$ drops below $\tau_{\min}$, and $\tau_{\mathrm{sum}} > 1$ is reduced, the second iteration may be more likely to construct OPT than the first. This effect is greatest when $\tau_t = \tau_{\min}$ and τ_{sum} is as large as possible, i. e., $1 + 1/n$; the probabilities of constructing OPT are then $\tau_{\min}/(1+1/n)$ and $\tau_{\min}/(1 + 1/n - \rho/n)$, i. e., the first iteration is $\frac{n+1-\rho}{n+1} \geq \frac{2-1/7}{2} = \frac{13}{14} \approx 0.92$ times as likely to construct OPT as the second, thus $p_f > 0.48 > 1/3$.

Pessimistically assuming that OPT is not constructed in the first iteration, so the pheromone value on the OPT edge evaporates during the first iteration, and lower-bounding the effect of the pheromone evaporation and reinforcement in the second iteration as 0, the pheromone value on the OPT edge is at least $\tau(1-\rho)$ during the third iteration, and hence the probability of constructing an ALT solution can be upper-bounded as $p_a \leq (1 - \tau_t(1-\rho)/\tau_{\mathrm{sum}}){\tau_{\max}}^{n-1} \leq (1 - \tau_t(1-\rho)/\tau_{\mathrm{sum}})(38/100)$ for $n > 15$.

Combining the three probability bounds, we consider the expected pheromone value τ_{t+1} on the edge corresponding to OPT, and then the distance X_{t+1}, which additionally decreases by $\frac{7}{2}\rho$ if OPT is constructed in either of the first two iterations, and ALT is not constructed in the third iteration:

$$\begin{aligned}
\mathrm{E}(\tau_{t+1} \mid C_t = 0) &\geq \tau_t(1-\rho)^3 + p_o\rho(p_f(1-\rho)^2 + (1-\rho) + (1-p_a)),\\
\mathrm{E}(X_t - X_{t+1} \mid C_t = 0) &\geq \mathrm{E}(\tau_{t+1} \mid C_t = 0) - \tau_t + p_o(1-p_a)\tfrac{7}{2}\rho\\
&\geq \rho\left(p_o\left(\tfrac{35}{6} - \tfrac{9}{2}p_a\right) - 3\tau_t\right) + \rho^2\left(3\tau_t - \tfrac{5}{3}p_o\right) - \rho^3\left(\tau_t + \tfrac{1}{3}p_o\right)\\
&= \rho\tau_t\left(\left(\tfrac{35}{6} - \tfrac{9}{2}p_a\right)b - 3 + \rho\left(3 - \tfrac{5}{3}b\right) - \rho^2\left(1 - \tfrac{1}{3}b\right)\right)\\
&= \rho\tau_t\left(\left(\tfrac{1237}{300} + \tfrac{171}{100}\tfrac{\tau_t}{\tau_{\mathrm{sum}}}\right)b - 3 + \rho\left(3 - \tfrac{5}{3}b - \tfrac{171}{100}\tfrac{\tau_t b}{\tau_{\mathrm{sum}}}\right) - \rho^2\left(1 - \tfrac{1}{3}b\right)\right),
\end{aligned}$$

where $b = \frac{2-\rho}{e\tau_{\mathrm{sum}}} - \tau_t\frac{1-\rho}{e^2{\tau_{\mathrm{sum}}}^2} \leq p_o/\tau_t$. By bounding $\frac{2}{e} > b > \frac{2}{e} - \frac{\tau_t}{e^2} - o(1)$,

$$\begin{aligned}
\mathrm{E}(X_t - X_{t+1} \mid C_t = 0) &= \rho\tau_t\left(\left(\tfrac{1237}{300} + 1.71\tau_t/\tau_{\mathrm{sum}}\right)b - 3 - O(\rho)\right)\\
&> \rho\tau_t\left(\left(\tfrac{1237}{300} + 1.71\tau_t/\tau_{\mathrm{sum}}\right)\left(2/e - \tau_t/e^2 - o(1)\right) - 3 - o(1)\right)\\
&= \rho\tau_t\left(\tfrac{1237}{150e} - 3 + \tau_t\left(\tfrac{3.42}{e\tau_{\mathrm{sum}}} - \tfrac{1237}{300e^2} - \tfrac{1.71\tau_t}{e^2\tau_{\mathrm{sum}}}\right) - o(1)\right)\\
&> \rho\tau_t\,(0.033 - o(1)) = \Omega(\rho(1 - X_t)),
\end{aligned}$$

by observing that for a sufficiently large n (and hence a τ_{sum} sufficiently close to 1), the expression multiplied with the inner τ_t is positive. □

Lemma 15 *Let the assumptions from Lemma 14 hold; recall that τ_t denotes the pheromone value on the edge corresponding to the* 0*-entry for the oscillating character after t oscillations. For $t \geq C(rn^2 + \alpha n)$, where $C > 0$ is some sufficiently large* *constant, and all* $\alpha \ge 0$ *it holds*

$$\Pr\left(\tau_t \le 1 - O\left(\frac{1}{n} + \frac{\log^2 n}{rn}\right)\right) \le e^{-\alpha} + e^{-\Omega(\log^2 n)}.$$

Proof We will apply Lemma 13, using the bounds from Lemma 14. Note that $0 \le X_t \le 1 - \frac{1}{n} - \frac{1}{rn} + \frac{7}{2}\rho \le 1 - \frac{1}{2n}$ since $\frac{1}{rn} \le \tau_t \le 1 - \frac{1}{n}$ and $\rho \le \frac{1}{7rn}$. We will consider the cases $C_t = 0$ and $C_t = 1$ separately. First, let $C_t = 0$. If $X_t \ge 1/n$, we get from Lemma 14 a drift $\mathrm{E}(X_t - X_{t+1} \mid X_t; X_t \ge 1/n) = \Omega(X_t\rho) = \Omega(1/(rn^2))$. We therefore set $x_{\min} := 0$, $a := \frac{1}{n}$, $b := a + \frac{\log^2 n}{rn}$, $x_{\max} = 1 - 1/(2n)$ and $h(x) := \frac{c_1}{rn^2}$ for some sufficiently small constant $c_1 > 0$ in Lemma 13. The choice of $h(x)$ results in $\mathrm{E}(X_t - X_{t+1} \mid X_t; X_t \ge a) \ge h(X_t)$.

By definition, $g(x) := \frac{xrn^2}{c_1}$ since $x_{\min} = 0$. Note that $X_t - X_{t+1} \le \frac{13}{2}\rho$ since three iterations can change τ_t by at most 3ρ and $C_t \le 1$ holds. This implies $g(X_t) - g(X_{t+1}) \le c_2rn^2\rho$ for some constant $c_2 > 0$. Hence, as $\rho = \Theta(1/(rn))$, we get $g(X_t) - g(X_{t+1}) \le c_5n$ for another constant $c_5 > 0$. An expansion of the exponential function will show for $\lambda := \frac{c_3}{n}$, where $c_3 > 0$ is a sufficiently small constant, that

$$\mathrm{E}(e^{-\lambda(g(X_t)-g(X_{t+1}))} \cdot \mathbb{1}\{X_t > a\} \mid X_t) \le \mathrm{E}(e^{-\lambda(g(X_t)-g(X_{t+1}))} \mid X_t; X_t > a) \le 1 - \frac{\lambda}{2} \le e^{-\lambda/2},$$

which then can be used to prove the lemma.

We supply the details for the expansion now. By setting $c_3 \le \frac{1}{c_5}$, we get $\lambda(g(X_t) - g(X_{t+1})) \le 1$. Using $e^{-x} \le 1 - x + x^2$ for $x \le 1$, we get for the moment-generating function (mostly omitting the conditions $X_t; X_t > a$ in expectations for readability)

$$\begin{aligned}\mathrm{E}\left(e^{-\lambda(g(X_t)-g(X_{t+1}))} \mid X_t; X_t > a\right) &\le 1 - \lambda\mathrm{E}\left(g(X_t) - g(X_{t+1})\right) + \lambda^2\mathrm{E}\left((g(X_t) - g(X_{t+1}))^2\right)\\ &\le 1 - \lambda\mathrm{E}\left(g(X_t) - g(X_{t+1})\right) + \lambda^2\mathrm{E}\left(|g(X_t) - g(X_{t+1})|\right)(c_5n)\end{aligned}$$

Since $\rho = \Theta(1/(rn))$, the third item of Lemma 14 states $\mathrm{E}(g(X_t) - g(X_{t+1})) \ge c_4X_tn$ for some constant $c_4 > 0$. We are left with an estimate for $\mathrm{E}(|\Delta|)$, where $\Delta := g(X_t) - g(X_{t+1})$. By the law of total probability (and again using $|\Delta| \le c_5n$),

$$\mathrm{E}(|\Delta|) \le \mathrm{E}(\Delta \mid \Delta > 0) + c_5n\Pr(\Delta < 0) \le c_6X_tn + c_5nc_7X_t \le (c_6 + c_7)c_5nX_t,$$

where we used the first and second item from Lemma 14 and introduced appropriate constants $c_6, c_7 > 0$ to cover the implicit constants from O-notation and the factor $1/c_1$ from $g(x)$.

Hence,

$$\begin{aligned}\mathrm{E}(e^{-\lambda(g(X_t)-g(X_{t+1}))}) &\le 1 - \lambda c_4X_tn + \lambda^2(c_6 + c_7)X_t(c_5n)^2\\ &\le 1 - \lambda c_4X_tn + \lambda\frac{c_3}{n}(c_6 + c_7)(c_5^2n)X_tn.\end{aligned}$$

Choosing $c_3 \le c_4/(2c_5^2(c_6 + c_7))$, the last bound is still bounded by $1 - \frac{\lambda}{2}\mathrm{E}(g(X_t) - g(X_{t+1}))$. Since $\mathrm{E}(g(X_t) - g(X_{t+1}) \mid X_t; X_t > a) \ge \mathrm{E}(X_t - X_{t+1} \mid X_t; X_t > a)/h(X_t) \ge h(X_t)/h(X_t) = 1$, the bound becomes at most $1 - \frac{\lambda}{2}$, which completes the analysis of the moment-generating function for $C_t = 0$ if $X_t > a$.

If $C_t = 1$, we can redo the above calculations analogously with $\mathrm{E}(g(X_t) - g(X_{t+1})) \geq c_4(1 - X_t)n$ and replace X_t with $1 - X_t$. We note that $1 - X_t \geq 1/(2n)$, hence still $\mathrm{E}(X_t - X_{t+1} \mid X_t; X_t \geq 1/n) = \Omega(1/(rn^2))$. In the estimation of $\mathrm{E}(|\Delta|)$, the events $\Delta < 0$ and $\Delta > 0$ are swapped. We get

$$\mathrm{E}(e^{-\lambda(g(X_t) - g(X_{t+1}))} \mid X_t; X_t > a) \leq 1 - \frac{\lambda}{2} \leq e^{-\lambda/2}$$

also in this case.

We are left with the case $X_t \leq a$ in Lemma 13. Pessimistically assuming the maximum change $(13/2)\rho$ of the X_t-values, we can bound

$$\mathrm{E}(e^{-\lambda(g(a) - g(X_{t+1}))} \cdot \mathbb{1}\{X_t \leq a\} \mid X_t) \leq e^{c_3 c_2 \rho r n} \leq D$$

for some constant $D > 0$. Applying Lemma 13 with $\beta := 1 - \lambda/2 \leq e^{-\lambda/2}$,

$$\Pr(X_t \geq b) \leq e^{-\frac{tc_3}{2n} + \frac{c_3}{n}\frac{rn^2}{c_1}} + \frac{D}{\lambda/2} \cdot e^{\lambda(g(a) - g(b))}$$

since $X_0 \leq 1$. Now, if $t = \frac{2rn^2}{c_1} + \frac{2\alpha n}{c_3}$ then $\Pr(X_t \geq b) \leq e^{-\alpha} + \frac{2D}{\lambda} e^{\lambda(g(a) - g(b))}$. The second term is $O(n) \cdot e^{-\Omega(\log^2(n))} = e^{-\Omega(\log^2(n))}$. Setting $C := \max\{\frac{2}{c_1}, \frac{2}{c_3}\}$ and noting that $X_t \geq b$ corresponds to $\tau_t \leq 1 - O(\frac{1}{n} + \frac{\log^2 n}{rn})$, the lemma follows. □

We remark here that the statement of Lemma 15 can most likely be strengthened to hold already for $t \geq C(rn \ln n + \alpha n)$ by using a different $h(x)$. However, since the bottleneck will be in the analysis of the time needed for phase 0, we are content with the present statement.

The following lemma takes the role of Lemma 4 in [9], which analyzes the transition from character i oscillating to character $i + 1$ oscillating. It applies to the case that the best-so-far solution at the end of phase i is not OPT_i but ALT_i despite the pheromones values favoring OPT_i. Then, when the new phase starts and the best-so-far is reevaluated, the fitness function will equal OneMax. However, it is very likely that MMAS* recovers quickly from this; more precisely, it will sample the solution OPT_i, which is in ALL_{i+1}, again before the pheromones have changed significantly. The lemma can be proven in the very same way as in [9]. In fact, the probability of setting a character being 0 in the best-so-far solution to 1, assuming saturated pheromones, will even be $1/(rn)$. This is less than the bound $1/n$ used in the proof from [9].

Lemma 16 *Let $\rho = \Theta(\frac{1}{rn})$. Assume for $i \in \{1, \ldots, n\}$ that the current-best solution is $0^{i-1}1^{n-i+1}$ and that the pheromones of the first $i - 1$ edges belonging to 0-entries as well as the last $n - i$ edges belonging to 1-entries all are $\tau_{max} = 1 - 1/n$. Finally, assume that the pheromone belonging to the i-th 0-entry is $1 - O((\log^2 n)/n)$. Then for all $c > 0$, MMAS* will sample $0^i 1^{n-i}$ within $O(\log n)$ iterations with probability $1 - O(n^{-c})$.*

Finally, Theorem 5 in [9] states a tail bound on the optimization time of classical MMAS* on OneMax, which is used in phase 0 of Maze, where the all-ones string is the first target. This theorem carries mostly over to our enlarged search space, see Theorem 18 below, except for that the modified lower pheromone border introduces a factor of r at two places. The following lemma is used for the analysis.

Lemma 17 *Assume there is a character whose value remains fixed in all best-so-far solutions of MMAS*. Then the pheromone values of the character will be saturated according to the best-so-far solution after at most* $\ln(rn)/\rho$ *steps.*

Proof The edges belonging to the r entries different from the best-so-far value each have a pheromone value of at most

$$\tau_{\max} \cdot (1-\rho)^t$$

or are already capped by their lower border after t steps, all of which by assumption reinforce the edge belonging to the best-so-far entry. Setting $t := \ln(rn)/\rho$, the expression becomes at most $1/(rn) = \tau_{\min}$. Since by Lemma 3 always $\tau_{\text{sum}} \geq 1$, the value for the best-so far entry must be at least $1 - r\tau_{\min} = \tau_{\max}$. □

Theorem 18 *For all* $c > 0$ *with probability* $1 - O(n^{-c})$*, MMAS* for the search space* $\{0, \ldots, r\}^n$ *optimizes* OneMax *in* $O(nr\log(rn)/\rho)$ *iterations and then saturates the all-ones string in pheromone.*

Proof We will use a fitness-level argument combined with an analysis of "freezing time" as commonly used in the analysis of ACO algorithms [12, page 125]. The number of characters being 1 in the best-so-far solution is non-decreasing over time. By Lemma 17, pessimistically assuming no update of the best-so-far, the pheromone values of every 1-entry must be saturated after at most $\ln(rn)/\rho$ steps. This applies to all characters simultaneously, also to the entries different from 1. Hence, pheromones values are saturated according to the best-so-far solution after a so-called freezing time of $O(\ln(rn)/\rho)$ steps (or an improvement is found before).

Given a current OneMax-value of $i \leq n-1$, the probability of finding an improvement in such a situation is at least

$$\binom{n-i}{1}{\tau_{\max}}^i \tau_{\min} \geq \frac{n-i}{ern} =: p_i$$

Hence, in the notation of Theorem 2 in [20], we have $n+1$ fitness levels $A_0, \ldots, A_n$ corresponding to the OneMax-values and corresponding probabilities of improvement (assuming saturated pheromones) given by p_i for $0 \leq i \leq n-1$. Now, $s = \sum_{i=0}^{m-1} 1/p_i^2 = O(rn)$, $h = \Omega(1/(rn))$ and $\sum_{i=0}^{n-1} 1/p_i = O(rn\log n)$. Hence, by setting $\delta = Ccrn\ln n$ for some constant $C > 0$ in the theorem, the time to reach the last level (without the freezing time) is $O(rn\log n)$ with probability $1 - n^{-c}$

On at most n levels, the pheromone values need to be saturated according to the best-so-far solution in order for the fitness-level argument to apply; one more saturation may be required for the final all-ones string. This accounts for a deterministic term of $O(n\ln(rn)/\rho)$ that has to be added to the time bound given by the fitness-level argument. Taking the two bounds together, the lemma follows. □

We note that the proof of Theorem 18 is the only place in our analysis where the strict >-selection for the update of the best-so-far solution by MMAS* is used. Otherwise, the arguments would hold for the algorithm using non-strict ≥-selection, which is often simply called MMAS in the literature.

Putting everything together, we obtain the following theorem, taking the role of Theorem 6 in [9]. Recall that t_0, the length of the so-called oscillation phase, is the number of iterations that a character is oscillating as OPT-OPT-ALT; however, the very first phase of length t_0, called phase 0, has objective function OneMax.

Theorem 19 *Given any $r > 0$, choose $\rho \le \frac{1}{7rn}$ and $\rho = \Omega(\frac{1}{rn})$. We say that MMAS* tracks the optimum of the* Maze *for the $(r+1)$-character alphabet if the best-so-far solution at the end of every oscillation phase has Hamming distance at most 1 to the optimum. Then for all $c > 0$ there is a constant k such that choosing $t_0 := kr^2n^2\ln(rn)$ makes MMAS* track the optimum with probability $1 - O(n^{-c})$.*

Proof We follow the argumentation in [9]. Let k' be the largest implicit constant in the bounds of Lemma 15, Lemma 16 and Theorem 18 for obtaining a failure probability of $O(n^{-c-1})$. Let $k = 3k'$. In the following, we assume that the events proved to hold with probability $1 - O(n^{-c-1})$ actually happen and call it a failure otherwise.

Since Maze equals OneMax within phase 0, we know from Theorem 18 that MMAS* finds the all-ones string and saturates all pheromones in the phase with probability $1 - O(n^{-c-1})$. The conditions of Lemma 15 hold for the start of phase 1, where the first character is oscillating. Setting $\alpha = n$, we get that the pheromone value on the corresponding 0-edge will be at least $1 - O(\log^2 n/(rn))$ by the end of the phase with probability $1 - o(n^{-c-1})$. The best-so-far solution by the end of phase 1 is guaranteed to be in ALL_1; however, it might be ALT_1, which does not belong to ALL_2.

We now analyze the transition to phase 2. According to Lemma 16, a solution from ALL_2 will be created in the first third of the phase with probability $1 - O(n^{-c-1})$. Within at most $O(rn\ln(rn))$ steps of the second third of the phase, MMAS* by Lemma 17 saturates all pheromones except for the oscillating character corresponding to the solutions from ALL_2. Now the conditions of Lemma 15 are satisfied. By the end of the final third of the phase, the pheromone value on the 0-edge for character 2 will be $1 - O(\log^2 n/(rn))$ with probability $1 - o(n^{-c-1})$. The subsequent phases are analyzed in the very same way as the second one.

By a union bound, the failure probability over all $n + 1$ phases is $O(n^{-c})$. □

6 Conclusions

We have revisited the analysis of evolutionary algorithms and ACO on the dynamic fitness function Maze [9]. First, we have shown that a (2+1) EA with a simple population diversity mechanism is able to track the optimum, which the (1+1) EA cannot. Subsequently, we have generalized this to a hierarchy result on strings over finite alphabets, where for given μ, there exists a Maze variant for which a population size of at least μ in a $(\mu+1)$ EA with genotype diversity is sufficient to track the optimum, whereas population size $\mu - 1$ causes the EA lose track of the optimum. Surprisingly, it turns out that a generalization of MMAS* to the larger state space does not require a population and is sufficient on all functions from the hierarchy. Along the way, we have introduced a variable drift theorem dealing with occupation probabilities, which allows for a more precise and simpler analysis of the pheromone values in MMAS* compared to [9]. As a subject for future research, it is interesting to study the benefits and the limitations of ACO on other dynamic problems.

References

1. Alba, E., Nakib, A., Siarry, P.: Metaheuristics for Dynamic Optimization. Studies in Computational Intelligence. Springer (2013)
2. Auger, A., Doerr, B. (eds.): Theory of Randomized Search Heuristics: Foundations and Recent Developments. World Scientific Publishing (2011)
3. Doerr, B., Pohl, S.: Run-time analysis of the (1+1) evolutionary algorithm optimizing linear functions over a finite alphabet. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '12), pp. 1317–1324. ACM Press (2012)
4. Droste, S.: Analysis of the (1+1) EA for a dynamically bitwise changing OneMax. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '03), pp. 909–921. Springer (2003)
5. Gunia, C.: On the analysis of the approximation capability of simple evolutionary algorithms for scheduling problems. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '05), pp. 571–578. ACM Press (2005)
6. Hajek, B.: Hitting and occupation time bounds implied by drift analysis with applications. Advances in Applied Probability **14**, 502–525 (1982)
7. Jansen, T., Schellbach, U.: Theoretical analysis of a mutation-based evolutionary algorithm for a tracking problem in the lattice. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '05), pp. 841–848. ACM Press (2005)
8. Jansen, T., Zarges, C.: Evolutionary algorithms and artificial immune systems on a bi-stable dynamic optimisation problem. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '14), pp. 975–982. ACM Press (2014)
9. Kötzing, T., Molter, H.: ACO beats EA on a dynamic pseudo-boolean function. In: Parallel Problem Solving from Nature-PPSN XII, pp. 113–122. Springer (2012)
10. Lehre, P.K., Witt, C.: General drift analysis with tail bounds (2013). ArXiv:1307.2559, `http://arxiv.org/abs/1307.2559`
11. Lissovoi, A., Witt, C.: MMAS vs. population-based EA on a family of dynamic fitness functions. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '14), pp. 1399–1406. ACM Press (2014)
12. Neumann, F., Witt, C.: Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity. Natural Computing Series. Springer (2010)
13. Nguyen, T.T., Yang, S., Branke, J.: Evolutionary dynamic optimization: A survey of the state of the art. Swarm and Evolutionary Computation **6**, 1–24 (2012)
14. Oliveto, P.S., Zarges, C.: Analysis of diversity mechanisms for optimisation in dynamic environments with low frequencies of change. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '13), pp. 837–844. ACM Press (2013)
15. Rohlfshagen, P., Lehre, P.K., Yao, X.: Dynamic evolutionary optimisation: An analysis of frequency and magnitude of change. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '09), pp. 1713–1720. ACM Press (2009)
16. Storch, T.: On the choice of the parent population size. Evolutionary Computation **16**(4), 557–578 (2008)
17. Stützle, T., Hoos, H.H.: Max-min ant system. Future Generation Computer Systems **16**(8), 889–914 (2000)
18. Sudholt, D.: Using markov-chain mixing time estimates for the analysis of ant colony optimization. In: Proceedings of the 11th Workshop on Foundations of Genetic Algorithms (FOGA '11), pp. 139–150. ACM Press (2011)
19. Sudholt, D., Thyssen, C.: Running time analysis of ant colony optimization for shortest path problems. J. Discrete Algorithms **10**, 165–180 (2012)
20. Witt, C.: Fitness levels with tail bounds for the analysis of randomized search heuristics. Information Processing Letters **114**(1), 38–41 (2014)