Runtime Analysis of a Co-Evolutionary Algorithm

Hevia Fajardo, Mario; Lehre, Per Kristian; Lin, Shishen

DOI:
10.1145/3594805.3607132

Citation for published version (Harvard):
Hevia Fajardo, M, Lehre, PK & Lin, S 2023, Runtime Analysis of a Co-Evolutionary Algorithm: Overcoming Negative Drift in Maximin-Optimisation. in *FOGA '23: Proceedings of the 17th ACM/SIGEVO Conference on Foundations of Genetic Algorithms.* FOGA: Foundations of Genetic Algorithms, Association for Computing Machinery (ACM), pp. 73–83, Foundations of Genetic Algorithms XVII, Potsdam, Germany, 30/08/23. https://doi.org/10.1145/3594805.3607132

Runtime Analysis of a Co-Evolutionary Algorithm: Overcoming Negative Drift in Maximin-Optimisation

Mario Alejandro Hevia Fajardo
School of Computer Science
University of Birmingham,
Birmingham, UK

Per Kristian Lehre
School of Computer Science
University of Birmingham,
Birmingham, UK

Shishen Lin
School of Computer Science
University of Birmingham,
Birmingham, UK

ABSTRACT

Co-evolutionary algorithms have found several applications in game-theoretic applications and optimisation problems with an adversary, particularly where the strategy space is discrete and exponentially large, and where classical game-theoretic methods fail. However, the application of co-evolutionary algorithms is difficult because they often display pathological behaviour, such as cyclic behaviour and evolutionary forgetting. These challenges have prevented the broad application of co-evolutionary algorithms.

We derive, via rigorous mathematical methods, bounds on the expected time of a simple co-evolutionary algorithm until it discovers a Maximin-solution on the pseudo-Boolean Bilinear problem. Despite the intransitive nature of the problem leading to a cyclic behaviour of the algorithm, we prove that the algorithm obtains the Maximin-solution in expected $O(n^{1.5})$ time.

However, we also show that the algorithm quickly forgets the Maximin-solution and moves away from it. These results in a large total regret of $\tilde{\Theta}(Tn^{1.5})$ after T iterations. Finally, we show that using a simple archive solves this problem reducing the total regret significantly.

Along the way, we present new mathematical tools to compute the expected time for co-evolutionary algorithms to obtain a Maximin-solution. We are confident that these tools can help further advance runtime analysis in both co-evolutionary and evolutionary algorithms.

CCS CONCEPTS

• **Theory of computation → Optimization with randomized search heuristics**.

KEYWORDS

Runtime analysis, Competitive coevolution, Maximin Optimisation

ACM Reference Format:
Mario Alejandro Hevia Fajardo, Per Kristian Lehre, and Shishen Lin. 2023. Runtime Analysis of a Co-Evolutionary Algorithm: Overcoming Negative Drift in Maximin-Optimisation. In *Proceedings of the 17th ACM/SIGEVO Conference on Foundations of Genetic Algorithms (FOGA '23), August 30-September 1, 2023, Potsdam, Germany*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3594805.3607132

FOGA '23, August 30-September 1, 2023, Potsdam, Germany

ACM ISBN 979-8-4007-0202-0/23/08.
https://doi.org/10.1145/3594805.3607132

1 INTRODUCTION

There is an increasing interest in optimisation problems that involve one or more adversaries, such as designing game-playing strategies, robust optimisation and Generative Adversarial Networks (GANs). Gradient-based methods have been used to solve these problems when the "payoff" function is differentiable. However, in many real-world scenarios, the payoff function is non-differentiable, that is, the derivatives of the function cannot be computed or do not exist. Non-differentiability can occur due to several reasons, such as when the domain is discrete or when the function is discontinuous. In such cases, gradient-based methods cannot be used, and alternative methods such as gradient-free algorithms or heuristic methods may be required to find an optimal solution. A promising approach in these settings is the use of competitive co-evolutionary algorithms (CoEAs[1]) [29] which are gradient-free algorithms. An early successful application of CoEAs was the work by Hillis [15] in which sorting networks were co-evolved with test cases. Other early examples are Axelrod et al. [3] and [24] independently using CoEAs to evolve effective strategies for the iterated prisoner's dilemma. Jensen, Branke and Rosenbusch [4, 17] studied CoEAs (gradient-free algorithms) on simple (continuous) adversarial problems empirically showing that with an appropriate fitness assignment method, CoEAs can obtain promising results.

More recently, there have been successful applications of CoEAs in co-evolutionary learning [25] and GANs [2, 8, 32]. CoEAs and GANs are adversarial models that share similarities. For example, they both are known to have pathological dynamics such as focusing, relativism, loss of gradient, mode collapse and cyclic behaviour [2, 9, 29, 31, 33]. To understand such dynamics, researchers have used basic adversarial problems for which simple algorithms present the aforementioned pathological behaviours. An example is the Maximin Bilinear problems (zero-sum games). Maximin Bilinear problems are a class of optimisation problems that involve maximising the minimum value of a bilinear function. Bilinear problems are often regarded as an important simple example for theoretically analyzing and understanding new algorithms and techniques [35]. These problems are known to be difficult and it is well-known that simultaneous gradient descent ascent does not converge [28]. However, Liang and Stokes [23], Mokhtari et al. [27], Zhang and Yu [35] showed that specialised gradient descent algorithms are able to get around these pathological phenomena and they can guarantee linear convergence to a *Nash equilibrium* on Minimax Bilinear problems. But these algorithms rely on a differentiable pay-off function, allowing the use of gradients.

[1] The name CoEAs is used for both competitive and cooperative co-evolutionary algorithms, in this work we refer only to competitive co-evolutionary algorithms

As mentioned before, in the case of discrete search spaces, there is no gradient, limiting the information the algorithms can use and making the task more difficult. Since CoEAs are gradient-free algorithms they can readily deal with discrete search spaces. Despite their promising results, CoEAs are not commonly used primarily due to a lack of proper understanding regarding their behavior.

Understanding the behaviour of CoEAs is considered to be more challenging than understanding classical evolutionary algorithms (EAs). While classical EAs select individuals for reproduction based on an external fitness function, CoEAs use a different approach. CoEAs typically involve two or more populations that compete with each other, and individuals are selected based on how well they "perform" against individuals from the other population(s) [29]. Hence, identifying the causes of pathological behaviours and designing CoEAs that can circumvent these issues is a challenging task. It requires a deep understanding of the complex interactions between the coevolving populations and the fitness landscape.

We argue that runtime analysis can be used to better understand CoEAs. Runtime analysis is an essential analytic tool in the study of time complexity for traditional EAs. It provides bounds for the number of fitness function evaluations (called runtime) [6]. Runtime analysis has helped identify and understand the relationships between algorithmic parameters and problem characteristics that determine the efficiency of evolutionary algorithms. In this work, we aim to expand the runtime analysis tool-set to CoEAs.

There are only a small number of rigorous runtime analysis of CoEAs. Jansen and Wiegand [16] analysed the runtime of a cooperative CoEA and a (1+1) EA on *separable* functions. The authors showed that problem separability cannot be used as a predictor of whether a cooperative CoEA performs better than the (1+1) EA. The first rigorous runtime analysis on competitive population-based CoEAs was made by Lehre [21]. The author proposed and theoretically analysed a population-based CoEA called PD-CoEA, showing that with the appropriate parameter values, it can efficiently find an ε-approximation of the Nash equilibria on some instances of a pseudo-Boolean Bilinear problem. Additionally, Lehre showed that for an incorrect parameter setting the PD-CoEA is inefficient on every problem, as long as the problem does not have too many optima. Following this work, Hevia Fajardo and Lehre [14] studied how fitness aggregation methods affect the performance of a $(1, \lambda)$ CoEA on a discrete lattice Bilinear problem. The authors showed that using the average of interactions as a fitness aggregation results in inefficient optimisation whilst using the worst case interaction as a fitness aggregation results in efficient optimisation.

1.1 Our Contributions

In this work, we analyse a simple CoEA that we name Randomised Local Search CoEA using Pairwise Dominance (RLS-PD, Algorithm 1) on some instances of the pseudo-Boolean Bilinear* problem[2] (formally defined in Section 2.1). Our main contribution is to demonstrate that utilizing a (1+1)-type CoEA can lead to undesired pathological behaviors, and it is essential to incorporate additional considerations such as archives in order to circumvent these issues.

First, we analyse the time it takes the RLS-PD to create a Nash equilibrium on instances of the Bilinear* problem where all Nash equilibria are at a Hamming distance of at most $O(\sqrt{n})$-bits from $n/2$-bits for both populations. Reaching a solution with $n/2$-bits is arguably the easiest problem setting, because the inherent genetic drift pushes the RLS-PD towards $n/2$-bits. Nevertheless, the setting where the Nash equilibria is moved by $\Theta(\sqrt{n})$ from $n/2$-bits can be challenging both for the RLS-PD to optimise and for us to analyse. This is because on Bilinear*, RLS-PD does not receive any signal towards or away from the optimum when selecting individuals. Therefore, it relies on the aforementioned genetic drift to approach the Nash equilibria. But once it draws near the Nash equilibria the genetic drift starts to work against the RLS-PD pushing it away. Despite this, we are able to show that the RLS-PD can find the Nash equilibrium efficiently on Bilinear*.

However, we also show that the RLS-PD does not converge to the Nash equilibrium. To illustrate this, we use the concept of regret and total regret. The concept of regret is commonly used in Maximin optimisation to comprehend the extent to which a decision's performance can be improved. In turn, the total regret provides a comprehensive measure of the overall deviation from the Maximin-solution in terms of fitness during a run, quantifying the extent to which a search-based algorithm's trajectory differs from a Maximin-solution. We give total regret bounds for the RLS-PD on Bilinear* showing that even after finding the Nash equilibrium, the algorithm quickly forgets it and stays away from it for a large proportion of the time resulting in a large total regret. The bounds for regret of RLS-PD are estimated by using Hajek's Occupation Time Bounds [12, Theorem 3.1].

Finally, we show that if an archive is used to retain promising solutions, the RLS-PD is able to reduce the total regret significantly. The reader may think that this is trivial, but due to the intransitivities of the Bilinear* problem an archive might prefer *bad* solutions over *good* solutions depending on what competing solutions are used during the comparison.

Due to the complexity of the coevolutionary dynamics, our analyses need analytical tools rarely used in runtime analysis, such as the Occupation Time Bound [12] and our newly developed variance drift theorem. The variance drift theorem builds on previous work where the variance of the process (or random fluctuations of the process) is used to overcome weak drift or zero drift tendencies [5, 7, 10, 11, 18]. In our case we extend previous variance drift theorems to allow even small negative drift as long as the variance is large enough and the distance to traverse is not too large. We are confident that this tool is of independent interest and can help further advance runtime analysis in both CoEAs and EAs.

We note that a similar drift analysis tool dealing with small negative drift in a small part of the optimisation have been used by Kötzing et al. [20]. The proof idea in [20] relies on a different drift transformation function which in turn results in different conditions needed to apply each theorem. Therefore, our theorem extends the toolbox of drift analysis.

A preliminary version of our results without the analyses on total regret (Section 4.1 and parts of Section 4.2) has been published as a poster at GECCO 2023.

[2] We use a small variation of the Bilinear function from [21] denoted by Bilinear*

2 PRELIMINARIES

We study a simple (1+1)-type of algorithm called Randomised Local Search with Pairwise Dominance (RLS-PD) shown in Algorithm 1. The algorithm initialises two solutions $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ uniformly at random from the strategy spaces $\mathcal{X}, \mathcal{Y} = \{0,1\}^n$. Being consistent with [21], we call x *predator* and y *prey*. Due to space constraints, we removed some proofs from the paper; the detailed proofs can be found in the supplementary material.

After the initialisation, in each iteration, the algorithm creates a new pair of solutions (x', y') by copying the parents (x, y) and flipping exactly one bit from either x or y. Later, the algorithm uses a dominance relation (Definition 2.1) to compare the created bit strings. If $(x', y') \succeq_g (x, y)$ then the new pair replaces the bit strings (x, y).

Definition 2.1 ([21])**.** Given a function $g = \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ and two pairs $(x_1, y_1), (x_2, y_2) \in \mathcal{X} \times \mathcal{Y}$, we say that (x_1, y_1) dominates (x_2, y_2) with respect to g, denoted $(x_1, y_1) \succeq_g (x_2, y_2)$, if and only if

$$g(x_1, y_2) \geq g(x_1, y_1) \geq g(x_2, y_1).$$

Algorithm 1 RLS-PD: Randomised Local Search with Pairwise Dominance.

```
Require: Maximin-objective function $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$.
Sample $x_1 \sim \mathrm{Unif}(\{0,1\}^n)$
Sample $y_1 \sim \mathrm{Unif}(\{0,1\}^n)$
for $t \in \{1, 2, \dots\}$ do
    Create $x', y' \in \{0,1\}^n$ by copying $x_t$ and $y_t$ and flipping exactly one bit chosen uniformly at random from either $x_t$ or $y_t$.
    if $(x', y') \succeq_g (x_t, y_t)$ then $(x_{t+1}, y_{t+1}) := (x', y')$
```

2.1 The Bilinear Problem

The pseudo-Boolean Bilinear problem (from now on we call it Bilinear) was proposed by Lehre [21] as a simple and well-defined class of Maximin-optimisation problems, where the Maximin-gradient is intransitive. These characteristics allow a better understanding of the behaviour of CoEAs via theoretical analysis, especially the pathological behaviours discussed before.

In this work, we use a slight variation of Bilinear denoted by Bilinear* because the original definition from [21] contains small plateaus where changing the decision of the predator x does not affect the best decision of the prey y. Although experimentally the small plateaus do not seem to have a large effect on the behaviour of the algorithm, it creates a problem in the theoretical analysis of the algorithm that to the best of our knowledge neither current nor our newly developed analytical tools are able to deal with. The Bilinear* function used in this work is defined for two parameters $\alpha, \beta \in (0,1)$ by

$$\textsc{Bilinear}^*_{\alpha,\beta}(x,y) := |y|_1\,|x|_1 - |y|_1\,\beta n - \alpha n\,|x|_1 + \frac{\max\{(\alpha n - |y|_1)^2, 1\}}{n^3} - \frac{\max\{(\beta n - |x|_1)^2, 1\}}{n^3}.$$

where for any bit string $z \in \{0,1\}^n$, $|z|_1 := \sum_{i=1}^n z^{(i)}$, that is, the number of 1-bits in z. Furthermore, we only consider the problem setting $\alpha = 1/2 \pm O(1/\sqrt{n})$ and $\beta = 1/2 \pm O(1/\sqrt{n})$. While this is arguably the easiest problem setting, it suffices to demonstrate that RLS-PD without an archive has unsatisfactory dynamics on Bilinear* (see Section 4).

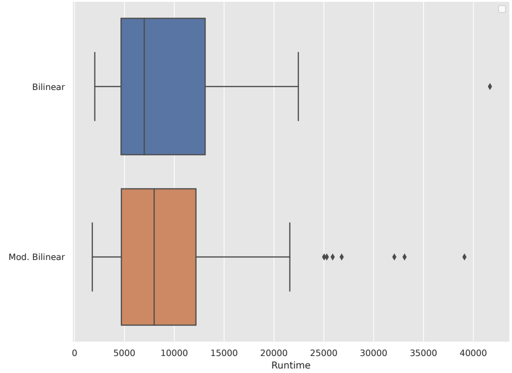


Figure 1: Average runtime over 100 runs of RLS-PD on the Bilinear function from [21] and the Bilinear* function studied here with $n = 1000$ and $\alpha = \beta = 0.56 \approx 0.5 + 2/\sqrt{n}$.

During our analysis, we divide the search space into four quadrants. We say that a pair of search points (x, y) is in:

- the first quadrant if $0 \leq |x|_1 < \beta n \wedge \alpha n \leq |y|_1 \leq n$,
- the second quadrant if $\beta n \leq |x|_1 \leq n \wedge \alpha n < |y|_1 \leq n$,
- the third quadrant if $\beta n < |x|_1 \leq n \wedge 0 \leq |y|_1 \leq \alpha n$, and
- the fourth quadrant if $0 \leq |x|_1 < \beta n \wedge 0 \leq |y|_1 < \alpha n$.

We also denote the set of Nash equilibria as OPT, that is, OPT := $\{(x,y) \mid |x|_1 = \beta n \wedge |y|_1 = \alpha n\}$. Note that this pure Nash equilibrium corresponds to the optimum solution of the Maximin optimisation problems. Our analysis and conclusions remain unchanged if we consider the variant of the problem.

We note that our definition of Bilinear* takes the same values as the definition of Bilinear from [21] almost everywhere and empirical analysis show that RLS-PD have a similar behaviour in both functions. That is, RLS-PD finds the optimum efficiently (Figure 1) but stays relatively far away from the optimum most of the time (Figure 2). The main purpose of the modification is to allow the theoretical analysis whilst maintaining the same general behaviour as the Bilinear function from [21]. We achieve this by adding a small gradient towards the Nash equilibrium when either $|x|_1 = \beta n$ or $|y|_1 = \alpha n$. In the following lemma, we show that the change in function values is minimal.

Lemma 2.2. *Let $h(x,y) := \max\{(\alpha n - |y|_1)^2, 1\}/n^3 - \max\{(\beta n - |x|_1)^2, 1\}/n^3$. Then, for any $(x_1, y_1), (x_2, y_2) \in \mathcal{X} \times \mathcal{Y}$ and any $\alpha, \beta \in [0,1]$*

$$|h(x_1, y_1) - h(x_2, y_2)| \leq \frac{2}{n} - \frac{2}{n^3}.$$

We now characterise the behaviour of RLS-PD. In the following lemma we show the required conditions for an iteration of RLS-PD to generate a pair of search points (x', y') for which $(x', y') \succeq_g (x, y)$.

Lemma 2.3. *Let $g := \textsc{Bilinear}^*$. Let (x, y) be the current pair of search points of Algorithm 1 and (x', y') be a pair of search points*

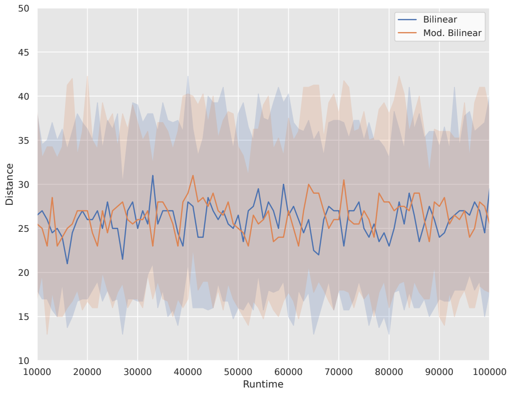


Figure 2: Median distance and interquartiles over 100 runs of RLS-PD on the Bilinear function from [21] and the Bilinear* function studied here with $n = 1000$ and $\alpha = \beta = 0.56 \approx 0.5 + 2/\sqrt{n}$.

created in Line 5 *of Algorithm 1. Then* $(x', y') \succeq_g (x, y)$ *if and only if any of the following conditions are true*

- $0 \le |x|_1 < \beta n \wedge \alpha n \le |y|_1 \le n$ *and*

$$(|y'|_1 = |y|_1 + 1 \wedge |x'|_1 = |x|_1) \\ \vee (|y'|_1 = |y|_1 \wedge |x'|_1 = |x|_1 + 1),$$

- $\beta n \le |x|_1 \le n \wedge \alpha n < |y|_1 \le n$ *and*

$$(|y'|_1 = |y|_1 - 1 \wedge |x'|_1 = |x|_1) \\ \vee (|y'|_1 = |y|_1 \wedge |x'|_1 = |x|_1 + 1),$$

- $\beta n < |x|_1 \le n \wedge 0 \le |y|_1 \le \alpha n$ *and*

$$(|y'|_1 = |y|_1 - 1 \wedge |x'|_1 = |x|_1) \\ \vee (|y'|_1 = |y|_1 \wedge |x'|_1 = |x|_1 - 1),$$

- $0 \le |x|_1 \le \beta n \wedge 0 \le |y|_1 < \alpha n$ *and*

$$(|y'|_1 = |y|_1 + 1 \wedge |x'|_1 = |x|_1) \\ \vee (|y'|_1 = |y|_1 \wedge |x'|_1 = |x|_1 - 1),$$

- $|x'|_1 = \beta n \wedge |y'|_1 = \alpha n$

We remark that Lemma 2.3 implies that as long as the algorithm does not reach the a Nash equilibria, it moves through the search space from quadrant to quadrant in order, that is first, second, third, fourth and back to first. This is because in the first quadrant only individuals $x'(y')$ with more one-bits than the parent $x(y)$ dominate their parents. In the second quadrant only individuals $x'(y')$ with more (less) one-bits than the parent $x(y)$ dominate the parents. In the third quadrant only individuals $x'(y')$ with fewer one-bits than the parent $x(y)$ dominate the parents. Finally, in the fourth quadrant only individuals $x'(y')$ with fewer (more) one-bits than the parent $x(y)$ dominate the parents.

2.2 Notation and Probability Estimates

Consider a filtration $\mathcal{F}_t$, we write $\mathbb{E}_t(\cdot) := \mathbb{E}(\cdot|\mathcal{F}_t)$. We use the following notation for RLS-PD (Algorithm 1) on Bilinear* throughout our analyses.

Definition 2.4. Let $M(x, y)$ be the Manhattan distance from a search point $(x, y) \in \mathcal{X} \times \mathcal{Y}$ to the optimum, that is, $M(x, y) := |n\beta - |x|_1| + |n\alpha - |y|_1|$. Let $M_t := M(x_t, y_t)$. For all $t \in \mathbb{N}$ we define:

$$p^+_{x,y} := \Pr(M_{t+1} > M_t \mid M_t = M(x, y));$$
$$p^-_{x,y} := \Pr(M_{t+1} < M_t \mid M_t = M(x, y)).$$

The comparisons between individuals depend on the number of 1-bits of the solutions x and y. To improve readability, we often use $i := |x_t|_1$ to denote the number of 1-bits in x_t and $j := |y_t|_1$ to denote the number of 1-bits in y_t.

Lemma 2.5. *For RLS-PD on Bilinear*, the quantities from Definition 2.4 are:*

$$p^+_{x,y} = \begin{cases} \frac{n-j}{2n} & 0 \le i < \beta n \wedge \alpha n \le j \le n \\ \frac{n-i}{2n} & \beta n \le i \le n \wedge \alpha n < j \le n \\ \frac{j}{2n} & \beta n < i \le n \wedge 0 \le j \le \alpha n \\ \frac{i}{2n} & 0 \le i \le \beta n \wedge 0 \le j < \alpha n \\ 1 & i = \beta n \wedge j = \alpha n \end{cases} \quad (1)$$

$$p^-_{x,y} = \begin{cases} \frac{n-i}{2n} & 0 \le i < \beta n \wedge \alpha n \le j \le n \\ \frac{j}{2n} & \beta n \le i \le n \wedge \alpha n < j \le n \\ \frac{i}{2n} & \beta n < i \le n \wedge 0 \le j \le \alpha n \\ \frac{n-j}{2n} & 0 \le i \le \beta n \wedge 0 \le j < \alpha n \\ 0 & i = \beta n \wedge j = \alpha n \end{cases} \quad (2)$$

2.3 Drift Theorems

In this section we include drift theorems that we will use during our analysis. We first state the "additive drift" theorem[3] which is due to [13].

Theorem 2.6 (Additive Drift [13]). *Let $(X_t)_{t\in\mathbb{N}}$, be a stochastic process, adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$, over some state space $S \subseteq \mathbb{R}$, and let $b, \delta_u, \delta_\ell > 0$. Then for $T_0 := \min\{t \mid X_t \le 0\}$ it holds:*

(i) If $E(X_t - X_{t+1} - \delta_u \,;\, X_t > 0 \mid \mathcal{F}_t) \ge 0$ and $X_t \ge 0$ for all for all $t \in \mathbb{N}$ then $E(T_0 \mid \mathcal{F}_0) \le \frac{X_0}{\delta_u}$.

(ii) If $E(X_t - X_{t+1} - \delta_\ell \,;\, X_t > 0 \mid \mathcal{F}_t) \le 0$ and $X_t \le b$ for all $t \in \mathbb{N}$, then $E(T_0 \mid \mathcal{F}_0) \ge \frac{X_0}{\delta_\ell}$.

Most drift theorems require the process to have in expectation a positive drift. However, sometimes the processes studied have a small negative drift pushing the process away from the target in expectation. In the following theorem, we present a transformation for such a process where the variance of the process can be used to counteract the small negative drift and allows us to apply drift theorems to show that the target is reached efficiently despite the negative drift. Doerr and Kötzing [5] use a similar idea to tackle the case of low drift. While compared with [5], we use a quadratic transformation instead of log-transformation to allow the $\Omega\left(\frac{1}{n}\right)$ negative drift before reaching the optimum.

[3] Following [22], the theorem is restated in terms of general stochastic processes instead of populations.

Theorem 2.7 (Variance drift transformation). *Let $(X_t)_{t\in\mathbb{N}}$ be a sequence of random variables with a finite state space $\mathcal{S} \subseteq \mathbb{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$, and let $T = \inf\{t \mid X_t \leq 0\}$. Furthermore, suppose that, given $b > 0$,*

(A$_1$) *there exist $\delta > 0$ such that for all $t \in \mathbb{N}$, it holds that*

$$\mathrm{E}_t\left((X_{t+1} - X_t)^2 - 2(X_{t+1} - X_t)(b - X_t) - \delta; t < T\right) \geq 0$$

Then, given that $Y_t = b^2 - (b - X_t)^2$ for all $t \in \mathbb{N}$

$$\mathrm{E}_t(Y_t - Y_{t+1} - \delta; t < T) \geq 0.$$

Proof. We define the process $Y_t = b^2 - (b - X_t)^2$ for all $t \in \mathbb{N}$. For all $t < T$ we determine the drift of Y_t:

$$\begin{aligned}
&\mathrm{E}_t(Y_t - Y_{t+1}; t < T)\\
&= \mathrm{E}_t\left((b - X_{t+1})^2 - (b - X_t)^2; t < T\right)\\
&= \mathrm{E}_t\left(b^2 - 2bX_{t+1} + X_{t+1}^2 - (b^2 - 2bX_t + X_t^2); t < T\right)\\
&= \mathrm{E}_t\left(X_{t+1}^2 - X_t^2 - 2b(X_{t+1} - X_t); t < T\right)\\
&= \mathrm{E}_t(X_{t+1}^2 - 2X_{t+1}X_t + X_t^2 - 2X_t^2\\
&\qquad + 2X_{t+1}X_t - 2b(X_{t+1} - X_t); t < T)\\
&= \mathrm{E}_t((X_{t+1} - X_t)^2 - 2X_t^2\\
&\qquad + 2X_{t+1}X_t - 2b(X_{t+1} - X_t); t < T)\\
&= \mathrm{E}_t((X_{t+1} - X_t)^2 + 2X_t(X_{t+1} - X_t)\\
&\qquad - 2b(X_{t+1} - X_t); t < T)\\
&= \mathrm{E}_t\left((X_{t+1} - X_t)^2 - 2(X_{t+1} - X_t)(b - X_t); t < T\right)\\
&\overset{(A_1)}{\geq} \delta.
\end{aligned}$$

□

Corollary 2.8 (Variance upper additive drift). *Let $a \leq 0$ and $b > 0$. Let $(X_t)_{t\in\mathbb{N}}$ be a sequence of random variables with a finite space state $\mathcal{S} \subseteq \mathbb{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$, and let $T = \inf\{t \mid X_t \leq 0\}$. Furthermore, suppose that, given $b > 0$,*

(A$_1$) *there exist $\delta > 0$ such that for all $t \in \mathbb{N}$, it holds that*

$$\mathrm{E}_t\left((X_{t+1} - X_t)^2 - 2(X_{t+1} - X_t)(b - X_t) - \delta; t < T\right) \geq 0$$

(A$_2$) *and for all $t \in \mathbb{N}$, it holds that $a \leq X_t \leq b$*

Then,

$$\mathrm{E}_0(T) \leq \frac{(b - \mathrm{E}_0(X_T))^2 - (b - X_0)^2}{\delta}.$$

Proof. We define the process $Y_t = b^2 - (b - X_t)^2$ for all $t \in \mathbb{N}$ again. Notice that $Y_t \leq 0$ is equivalent to $X_t \leq 0$ by using A_2 condition. Then

$$T = \inf\{t \mid X_t \leq 0\} = \inf\{t \mid Y_t \leq 0\}.$$

We apply a version of the additive drift theorem from [19, Theorem 7] on the stochastic process Y_t and get the claimed upper bounds for the expected runtime. □

When applying drift analysis, we need to use potential functions that describe the progress of the algorithm towards a target. Sometimes different parts of the process are better described by different potential functions, but we need to choose only one potential function to describe the whole process because of the requirements imposed by the drift theorems. In Lemma 2.9 we show a new and intuitive way for which we can join two potential functions allowing us to use more than one potential function.

Lemma 2.9 (Joining potential functions). *Let $(X_t)_{t\in\mathbb{N}}$ be a sequence of random variables with a finite state space $\mathcal{S} \subseteq \mathbb{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$, $\varepsilon > 0$, $0 \leq x_{\min} \leq a \leq x_{\max}$ and let T be a stopping time. Additionally, let $\mathcal{A} = [x_{\min}, a + \varepsilon]$ and $\mathcal{B} = [a - \varepsilon, x_{\max}]$ be real intervals that contain at least all values x that, for all $t \leq T$, any X_t can take. Furthermore, suppose that,*

(B$_1$) *for all $t < T$, it holds that $|X_t - X_{t+1}| \leq \varepsilon$,*

(B$_2$) *there exist a function $f\colon \mathcal{A} \to \mathbb{R}$ and a function $\delta_1\colon \mathcal{A} \setminus (a, a + \varepsilon] \to \mathbb{R}^+$ such that for all $t < T$, it holds that*

$$\mathrm{E}_t(f(X_t) - f(X_{t+1}) - \delta_1(X_t); t < T \wedge X_t \leq a) \geq 0,$$

(B$_3$) *there exist a function $g\colon \mathcal{B} \to \mathbb{R}$ and a function $\delta_2\colon \mathcal{B} \setminus [a - \varepsilon, a) \to \mathbb{R}^+$ such that for all $t < T$, it holds that*

$$\mathrm{E}_t(g(X_t) - g(X_{t+1}) - \delta_2(X_t); t < T \wedge X_t > a) \geq 0,$$

(B$_4$) *for all $x \in [a, a + \varepsilon]$, $f(x) \geq g(x)$ and for all $x \in [a - \varepsilon, a]$, $g(x) \geq f(x)$.*

Then, for all $t < T$, it holds that

$$\mathrm{E}_t(h(X_t) - h(X_{t+1}) - \delta(X_t); t < T) \geq 0$$

with

$$h(x) = \begin{cases} f(x) & \text{if } x \leq a \\ g(x) & \text{if } x > a \end{cases} \qquad \delta(x) = \begin{cases} \delta_1(x) & \text{if } x \leq a \\ \delta_2(x) & \text{if } x > a. \end{cases}$$

We will also need the following drift theorem from [18] (see also Corollary 4.1 in [34]).

Theorem 2.10 ([18]). *Let $(X_t)_{t\in\mathbb{N}}$ be random variables over $\mathbb{R}$, each with finite expectation and let $m > 0$. With $T := \inf\{t \geq 0 \mid X_t \geq m\}$, we denote the random variable describing the earliest point that the random process exits the interval $[0, m)$. Suppose there are $\varepsilon, c > 0$ such that for all t,*

1. $\mathrm{E}(X_{t+1} - X_t - \varepsilon; T > t \mid X_0, \ldots, X_t) \leq 0$

2. $|X_t - X_{t+1}| < c$

3. $X_0 \leq 0$

Then, for all $s < m/(2\varepsilon)$, $\Pr(T < s) \leq \exp\left(-\frac{m^2}{8c^2 s}\right)$.

3 RLS-PD SOLVES BILINEAR* EFFICIENTLY

In this section we analyse the expected runtime of RLS-PD on Bilinear*.

First, we show that RLS-PD optimises Bilinear* in $O(n^{1.5})$ expected function evaluations. The following theorem states the main result of this section.

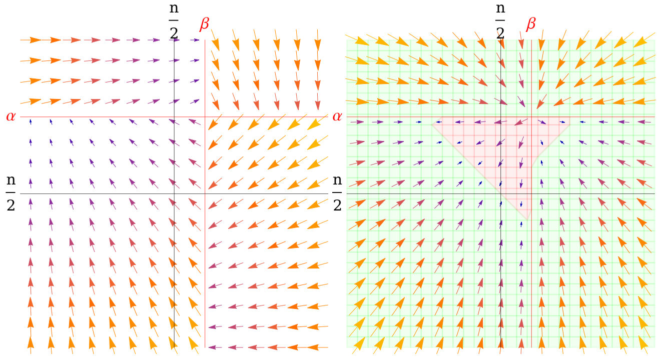
Theorem 3.1. *Consider $\alpha \in [1/2 - A/\sqrt{n}, 1/2 + A/\sqrt{n}]$ and $\beta \in [1/2 - B/\sqrt{n}, 1/2 + B/\sqrt{n}]$, where $A, B > 0$ are constants and $3(A + B)^2 \leq 1/2 - \delta'$ for some constant $\delta' > 0$. Define $T := \inf\{t \mid (x_t, y_t) \in$*

OPT}, *where* (x_t, y_t) *are the current solutions of RLS-PD. Consider RLS-PD on* BILINEAR$^*{}_{\alpha,\beta}$*. Then, for any initial search points* (x_0, y_0), $\mathrm{E}(T) = O(n^{1.5})$.

The proof of Theorem 3.1 relies on the optimum being located roughly at $n/2$ 1-bits for both predator and prey. The intuition behind the proof is as follows. When the current solutions are far from the optimum, it is easier to flip bits that reduce the Manhattan distance of the current points to the optimum than flipping bits that increase it because of the genetic drift inherent to the mutation operator. For example, if the current search points are in the first quadrant with $|x|_1 = o(1)$ and $|y|_1 = n - o(1)$, then there are much more 1-bits in y than in x, hence the probability of flipping a 0-bit in y (increasing the Manhattan distance) is smaller than the probability of flipping a 0-bit in x (reducing the Manhattan distance). In Figure 3 (a) we visualise this behaviour in detail for αn and βn different than $n/2$.

We use this behaviour and compute the expected decrease in Manhattan distance to the optimum using drift analysis (Lemma 3.2) to show that the algorithm approaches the optimum efficiently when the current Manhattan distance is sufficiently large. Figure 3 (b) shows in green the regions where this is true.

Once the current solution is near the optimum, if $\alpha n \neq n/2$ or $\beta n \neq n/2$ the genetic drift can work against the algorithm, increasing the Manhattan distance in expectation (as seen in the pink region of the right hand-side plot in Figure 3 (b)). Despite this, with the use of our new variance drift theorem (Theorem 2.7), we can show that the expected increase in Manhattan distance is negligible and the variance of the mutation operator is enough to guide the current solution towards the optimum efficiently.



(a) (b)

Figure 3: Expected change in position (a) and expected change in Manhattan distance (b) for $\alpha \neq n/2$ and $\beta \neq n/2$.

Lemma 3.2. *Consider RLS-PD on* BILINEAR* *as in Theorem 3.1. Define* $T := \inf\{t \mid M_t = 0\}$*, then for every generation* $t < T$

$$\mathrm{E}\left(M_t - M_{t+1} - \frac{M_t - (A+B)\sqrt{n}}{2n}; t < T \mid M_t\right) \geq 0.$$

From Lemma 3.2 we can see that when $M_t \leq (A+B)\sqrt{n}$ the drift cannot be guaranteed to be positive, therefore, we need a different approach to show that we can reach the optimum. For this purpose, we use the variance drift theorem (Theorem 2.7).

Lemma 3.3. *Consider RLS-PD on* BILINEAR* *as in Theorem 3.1. Define* $T := \inf\{t \mid M_t = 0\}$*. Define the process* $(Y_t)_{t\in\mathbb{N}}$ *with* $Y_t = b^2 - (b - M_t)^2$ *and* $b = 2(A+B)\sqrt{n} + 1$*, then there exists a constant* $\delta_1 > 0$ *for which*

$$\mathrm{E}_t(Y_t - Y_{t+1} - \delta_1; t < T \wedge M_t \leq b) \geq 0$$

Finally we use Lemma 2.9 to combine Lemmas 3.2 and 3.3 and show that the algorithm reaches the optimum in $O(n^{1.5})$.

Proof of Theorem 3.1. We aim to use Lemma 2.9 to combine Lemmas 3.2 and 3.3, therefore we need to show that the conditions of the lemma hold. We consider the values $X_t = M_t$, $x_{\min} = 0$, $x_{\max} = n + b$, $a = 2(A+B)\sqrt{n}$ and $\varepsilon = 1$ for the variables in Lemma 2.9. The algorithm flips at most one bit per iteration, hence $|X_t - X_{t+1}| \leq 1$ which meets Condition (B$_1$). Conditions (B$_2$) and (B$_3$) hold for $f(x) = b^2 - (b-x)^2$ and $g(x) = b^2 - (b-x)$ with $b = 2(A+B)\sqrt{n} + 1$ by Lemmas 3.3 and 3.2 respectively. We note that Lemma 3.3 pertains only the Manhattan distance M, but since M and $g(M)$ only differ by an additive constant, Lemma 3.3 also applies for $g(M)$. Finally, for Condition (B$_4$) we compute

$$f(x) - g(x) = (b-x) - (b-x)^2 = (b-x)(x - (b-1)). \quad (3)$$

For $x = a = b - 1$ and $x = a + 1 = b$ this is equal to 0. For $x \in (a, a+1) = (b-1, b)$ both terms in Equation (3) are positive, hence, $f(x) \geq g(x)$ for all $x \in [a, a+1]$. For $x \in [a-1, a) = [b-2, b-1)$ the first term in Equation (3) is positive and the second term is negative, hence, $g(x) \geq f(x)$ for all $x \in [a-1, a]$.

Given that all conditions in Lemma 2.9 hold we know that the drift in the potential function

$$h(M_t) = \begin{cases} b^2 - (b - M_t)^2 & \text{if } M_t \leq 2(A+B)\sqrt{n} \\ b^2 - (b - M_t) & \text{if } M_t > 2(A+B)\sqrt{n} \end{cases}$$

is lower bounded by the function

$$\delta(M_t) = \begin{cases} \delta_1 & \text{if } M_t \leq 2(A+B)\sqrt{n} \\ \frac{M_t - (A+B)\sqrt{n}}{2n} & \text{if } M_t > 2(A+B)\sqrt{n}. \end{cases}$$

Since $h(M_t) = 0$ implies $M_t = 0$ we use $\delta(M_t)$ to find an upper bound on $\mathrm{E}(T)$ as follows. First we note that $\frac{M_t - (A+B)\sqrt{n}}{2n} \geq 1/(2\sqrt{n})$ for all $M_t > 2(A+B)\sqrt{n}$ and for a sufficiently large n, $\delta_1 \geq 1/(2\sqrt{n})$. Then using additive drift (Theorem 2.6) with $h(M_0) = O(n)$ we obtain

$$\mathrm{E}(T) \leq \frac{O(n)}{1/(2\sqrt{n})} = O(n^{1.5}).$$

□

Corollary 3.4. *Consider* $\alpha = \beta = 1/2$*. Define* $T := \inf\{t \mid (x_t, y_t) \in$ OPT}*, where* (x_t, y_t) *are the current solutions of RLS-PD. Consider RLS-PD on* BILINEAR$^*{}_{\alpha,\beta}$*. Then, for any initial search points* (x_0, y_0), $\mathrm{E}(T) = O(n \log(n))$.

3.1 Lower Bounds

In this section we show that the algorithm needs at least a linear number of function evaluations to reach a Nash equilibrium. We start by giving an upper bound on the expected change in Manhattan distance.

Lemma 3.5. *Consider RLS-PD on* Bilinear* *as in Theorem 3.6. Define* $T := \inf\{t \mid M_t = 0\}$, *then for every iteration* $t \in \mathbb{N}$,

$$\mathrm{E}\left(M_t - M_{t+1} - \frac{M_t + (A+B)\sqrt{n}}{2n}; t < T \mid M_t\right) \leq 0.$$

With Lemma 3.5 we can now present the main theorem of this section.

Theorem 3.6. *Consider any* $\alpha \in [1/2 - A/\sqrt{n}, 1/2 + A/\sqrt{n}]$ *and* $\beta \in [1/2 - B/\sqrt{n}, 1/2 + B/\sqrt{n}]$. *Define* $T := \inf\{t \mid (x_t, y_t) \in \mathrm{OPT}\}$, *where* (x_t, y_t) *are the solutions of RLS-PD in iteration* $t \in \mathbb{N}$ *when applied to* $\textsc{Bilinear}^*_{\alpha,\beta}$. *In particular, if* $M(x_0, y_0) \geq C\sqrt{n}$, $A, B = \Theta(1)$ *and* $C > 1$, *then the lower bound is of order* $\Theta(n)$ *where* M *is defined in Definition 2.4.*

Proof. For a lower bound, we only count the iterations where $M_t := M(x_t, y_t) < C\sqrt{n}$. For all $t \in \mathbb{N}$ let $X_t := M_t$ and define $\delta_\ell := \frac{A+B+C}{2\sqrt{n}}$ and $b := C\sqrt{n}$. By Lemma 3.5, the conditions of Theorem 2.6 are satisfied, which implies

$$\mathrm{E}(T) \geq \frac{C\sqrt{n}}{\delta_\ell} = \frac{2Cn}{A+B+C}.$$

□

4 RLS-PD FORGETS THE NASH EQUILIBRIUM

Despite the algorithm finding a Nash equilibrium efficiently, the inherent characteristics of the function cause the algorithm not only to forget the Nash equilibrium found but also move away from it by a distance $\Omega(\sqrt{n})$ in $O(n)$ iterations. This is shown in the following theorem.

Theorem 4.1. *Let* $\alpha = 1/2 \pm A/\sqrt{n}$ *and* $\beta = 1/2 \pm B/\sqrt{n}$, *where* $A, B > 0$ *are constants. Define* $T := \inf\{t \mid M_t \geq (A+B)\sqrt{n}\}$, *where* M_t *is the current Manhattan distance to the set* OPT. OPT *is the set of the current solutions of RLS-PD. Consider RLS-PD on* $\textsc{Bilinear}^*_{\alpha,\beta}$. *Then, for any initial search points* (x_0, y_0), $\mathrm{E}(T) = O(n)$.

Proof. If the initial search points (x_0, y_0) have $M_0 > (A+B)\sqrt{n}$ then, $T = 1 = O(n)$. Hence from now on we assume $M_0 \leq (A+B)\sqrt{n}$.

Notice that for $M_t \leq (A+B)\sqrt{n}$ the $\mathrm{E}_t(M_t - M_{t+1})$ can be positive or negative, in order to cope with this, we use the variance drift theorem (Corollary 2.8) to show the desired expected runtime. Let us set $a = 0, b = (A+B)\sqrt{n}$ and $X_t = b - M_t$ in conditions of Corollary 2.8. Since $M_t \geq 0$ then $X_t \leq b$. Additionally, we note that $|M_t - M_{t+1}| = 1$, and if $M_{t+1} \geq b$, then from the definition of T we are done. Hence, for all $t \leq T$, $0 \leq X_t \leq b$ meeting Condition (A$_2$).

Now, we check Condition (A$_1$). We first need to compute the second moment of $(X_{t+1} - X_t)$. For every generation $t < T$, we have

$$\begin{aligned}\mathrm{E}_t\left((X_{t+1} - X_t)^2\right) &= \mathrm{E}_t\left((M_t - M_{t+1})^2\right)\\ &= p^+_{x,y}(M_t - (M_t + 1))^2\\ &+ p^-_{x,y}(M_t - (M_t - 1))^2\\ &= p^+_{x,y} + p^-_{x,y}.\end{aligned}$$

by Lemma 2.5, we write the expression explicitly as

$$\begin{aligned}&\mathrm{E}_t\left((X_{t+1} - X_t)^2\right)\\ &= \begin{cases} \frac{n-j}{2n} + \frac{n-i}{2n} & 0 \leq i < \beta n, \alpha n \leq j \leq n,\\ \frac{n-i}{2n} + \frac{j}{2n} & \beta n \leq i \leq n, \alpha n < j \leq n,\\ \frac{i}{2n} + \frac{j}{2n} & \beta n < i \leq n, 0 \leq j \leq \alpha n,\\ \frac{i}{2n} + \frac{n-j}{2n} & 0 \leq i \leq \beta n, 0 \leq j < \alpha n,\\ 1 & i = \beta n, j = \alpha n. \end{cases}\\ &\geq \frac{1}{2} - \frac{A+B}{\sqrt{n}}, \end{aligned} \tag{4}$$

where the last inequality follows by bounding i and j as done in the proof of Lemma 3.3 but using $b = (A+B)\sqrt{n}$ instead of $b = 2(A+B)\sqrt{n} + 1$. Now we note that $\mathrm{E}_t(X_{t+1} - X_t) = \mathrm{E}_t(M_t - M_{t+1})$ and compute $\mathrm{E}_t(M_t - M_{t+1})$.

$$\begin{aligned}&\mathrm{E}_t(M_t - M_{t+1})\\ &= p^-_{x,y} - p^+_{x,y}\\ &= \begin{cases} \frac{j-i}{2n} & 0 \leq i < \beta n, \alpha n < j \leq n\\ \frac{j+i-n}{2n} & \beta n < i \leq n, \alpha n < j \leq n\\ \frac{i-j}{2n} & \beta n < i \leq n, 0 \leq j < \alpha n\\ \frac{n-j-i}{2n} & 0 \leq i < \beta n, 0 \leq j < \alpha n\\ 1 & i = \beta n, j = \alpha n \end{cases}\end{aligned}$$

With the same arguments used in the proof of Lemma 3.2 we obtain

$$\mathrm{E}_t(M_t - M_{t+1}) \leq \frac{M_t + (A+B)\sqrt{n}}{2n}.$$

Hence,

$$\begin{aligned}\mathrm{E}_t(2(X_{t+1} - X_t)(b - X_t)) &\leq 2M_t \cdot \frac{M_t + (A+B)\sqrt{n}}{2n}\\ &= M_t \cdot \frac{M_t + b}{n} \leq \frac{b-1}{n}\\ &\leq \frac{(A+B)}{\sqrt{n}}\end{aligned} \tag{5}$$

Joining Equations (4) and (5) yields

$$\begin{aligned}&\mathrm{E}_t\left((X_{t+1} - X_t)^2 - 2(X_{t+1} - X_t)(b - X_t); t < T\right)\\ &\geq \frac{1}{2} - \frac{2(A+B)}{\sqrt{n}}\end{aligned}$$

For sufficiently large n and $\delta := 1/4$ the Condition (A$_1$) in Corollary 2.8 is met. Hence, the expected runtime is

$$E(T) \leq \frac{(b-a)^2 - (b - X_0)^2}{\delta} \leq \frac{b^2}{\delta} = O(n).$$

□

4.1 Large Regret

The Maximin-regret is commonly used as a performance measure for Maximin optimisation. This is because it allows us to understand how much the performance of a decision can be improved. Following [1], we define the Maximin-regret of Bilinear* as

$$
\begin{aligned}
r(x) :=& \textsc{Bilinear}^*(x_0, y_0) - \min_y \textsc{Bilinear}^*(x, y) \\
=& \begin{cases} -\alpha n\beta n + \alpha ni - n(i - \beta n) \\ \qquad + \frac{(\alpha n - n)^2 - (\beta n - i)^2}{n^3} & i < \beta n \\ -\alpha n\beta n + \alpha ni + \frac{(\alpha n)^2 - (\beta n - i)^2}{n^3} & i > \beta n \\ 0 & i = \beta n \end{cases} \\
=& \begin{cases} (i - \beta n)n(\alpha - 1) + O(\frac{1}{n}) & i < \beta n \\ (i - \beta n)n\alpha + O(\frac{1}{n}) & i > \beta n \\ 0 & i = \beta n. \end{cases}
\end{aligned}
$$

where $(x_0, y_0) \in \text{OPT} = \{(x, y) \mid |x|_1 = \beta n \wedge |y|_1 = \alpha n\}$.

In turn, the total Maximin-regret quantifies the extent to which a search-based algorithm's trajectory deviates from a Maximin-solution in terms of fitness, providing a comprehensive measure of overall deviation. A large total Maximin-regret imply that the algorithm does not converge to a Maximin-solution. We define the total Maximin-regret of a run as:

$$
R = \sum_{t=1}^{T} r(x_t),
$$

for any $T \geq 1$. For algorithms which in iteration t maintain an archive of m_t solutions $x_t^{(1)}, \dots, x_t^{(m_t)}$, we define the total regret of the run as $R = \sum_{t=1}^{T} \min_{i \in [m_t]} r(x_t^{(i)})$.

Comparing this regret with the regret for the definition of Bilinear in [21], we have at most an additional term $O(1/n)$. In the following analysis, we state the cumulative regret in asymptotic notation where lower-order terms are not shown. Hence, we will proceed as we analyse the regret for the original variant of Bilinear defined in [21].

4.1.1 Lower bound.

We first prove a lower bound on the regret. Our analysis follows the intuition that the search point induced by the algorithm will quickly reach within Manhattan distance $(A + B)\sqrt{n}$ from the optimum, however from that point, it will take $\Omega(n)$ iterations to reduce the Manhattan distance further to $(A + B)\sqrt{n}/2$. Within this time interval, the algorithm will "cycle" multiple times around the optimum. The lower bound on the regret corresponds to the regret accumulated within this time interval.

We start by showing that the Manhattan distance will remain within $\Theta(\sqrt{n})$ during a time interval of $\Theta(n)$ iterations.

Lemma 4.2. *Assume that $A + B \geq 1$. Given that M_t from Definition 2.4, there exists a constant $c > 0$ such that for any starting point (x_1, y_1), $\Pr\left(\bigwedge_{t=cn}^{cn+n/16} M_t \geq (A + B)\sqrt{n}/2\right) \geq 1/5$.*

As defined above, the Maximin-regret depends only on x and is independent of y. Hence, the Manhattan-distance $M_t = ||x| - \beta n| + ||y| - \alpha n|$ cannot be used directly to bound the regret. Instead, we will show that during $\Omega(n)$ iterations, the algorithm will "cycle" around OPT $\Omega(\sqrt{n})$ times. Given the lower bound on M_t, we must have $||x| - \beta n| = \Omega(\sqrt{n})$ during $\Omega(\sqrt{n})$ iterations of each cycle. Taking into account that the algorithm requires in expectation $\Theta(n)$ iterations to reduce the Manhattan distance from $(A + B)\sqrt{n}$ to $(A + B)\sqrt{n}/2$, this will be sufficient to prove an almost tight lower bound on the regret.

Theorem 4.3. *Let $\alpha = 1/2 \pm A/\sqrt{n}$ and $\beta = 1/2 \pm B/\sqrt{n}$, where $A, B > 0$ are constants. The expected total regret of RLS-PD on $\textsc{Bilinear}^*_{\alpha,\beta}$ is $\Omega(Tn^{1.5})$, where T is the number of iterations for which the algorithm has run.*

Proof. For a lower bound on the regret, we modify the process M_t so that $M_t \leq (A + B)\sqrt{n}$ for all $t \geq cn$, where c is as defined in Lemma 4.2.

We consider a phase of $T := cn + n/16$ iterations, and call the phase a failure if the event $\bigwedge_{t=cn}^{cn+n/16} M_t \geq (A + B)\sqrt{n}/2$ does not occur. By Lemma 4.2, the probability of this failure is at most 4/5.

We call a step "relevant" if the total number of 1-bits in x and y changes by 1. By Lemma 2.5, the probability of a relevant step is at least $\frac{n/2 - (A+B)\sqrt{n}}{n} > 1/3$. Hence, by a Chernoff bound [26], the probability of less than $n/48$ relevant steps during $n/16$ iterations is $1 - e^{-\Omega(n)}$, and we call the phase a failure otherwise.

Assuming no failure, it follows that

$$
(A + B)\sqrt{n}/2 \leq M_t \leq (A + B)\sqrt{n} \tag{6}
$$

for all $cn \leq t \leq cn + n/16$.

We call a "cycle" a sub-phase starting from an iteration t to iteration $t' > t$ such that at iteration t, it holds $|y| = \alpha n$ and $|x| > \beta n$, at iteration t' it holds $|y| > \alpha n$ and $|x| > \beta n$, and at iteration $t' + 1$, it holds $|y| = \alpha n$ and $|x| > \beta n$. Therefore, each cycle starts at the third quadrant and generation t' is the last generation where the algorithm transitions from the second quadrant to the third quadrant. We remark that due to the dominance relation the algorithm can only accept new individuals if they follow the direction of a cycle, that is third quadrant followed by the fourth quadrant, then first quadrant and finally the second quadrant.

The number of one-bits in x and y cannot both increase or both decrease, within one step. Hence, since (6) holds from iteration T until the end of the phase, the duration of each cycle is at most $O(\sqrt{n})$ relevant steps.

Furthermore, (6) also implies that during each cycle, there must be $\Theta(\sqrt{n})$ relevant steps where $||x|_1 - \beta n| \geq (A + B)\sqrt{n}/4$. During such a step, the regret is $\Omega(n\sqrt{n})$. Within $n/48$ relevant steps, there must have been at least $\Omega(\sqrt{n})$ cycles. Hence, between iteration cn and iteration $cn + n/16$, there must be at least $\Omega(n)$ relevant steps where the regret is $\Omega(n\sqrt{n})$.

Overall, unless a failure occurs, the accumulative regret during T iterations is $\Omega(n^{2.5}) = \Omega(Tn^{1.5})$. The result now follows by taking into account that no failure occurs with constant probability. □

4.1.2 Upper bound.

For the upper bound on the regret, our analysis will follow the intuition that the algorithm has positive drift towards the optimum when the Manhattan distance is above $(A + B)\sqrt{n}$. We will prove that it is unlikely that the algorithm spends significant amount of by a log-factor above this distance. The analysis relies on occupation-time bounds which have rarely been applied in theory of evolutionary algorithms (see e.g. [30]).

We first lower bound the probability of an improving step.

Lemma 4.4. *Let $\alpha = 1/2 \pm A/\sqrt{n}$ and $\beta = 1/2 \pm B/\sqrt{n}$, where $A, B > 0$ are constants. Let $\delta := \frac{1}{\sqrt{n}-1}$, $a := (A + B)\sqrt{n} + ((1 + \delta)^2 - 1) \cdot (n/2 + (A + B)\sqrt{n})$ and $M_t \geq a$. Then, $p^-_{x,y} \geq p^+_{x,y}(1 + \delta)^2$ and for sufficiently large n $p^-_{x,y} \geq 1/4 - (A + B)/\sqrt{n} \geq 1/8$.*

We then derive an occupation-time bound, applying a result from [12]. Informally, the statement implies that the algorithm will not spend too much time at a Manhattan distance significantly larger than $(A+B)\sqrt{n}$.

Lemma 4.5. *Assume n large enough (i.e. $n \geq 5$), let $\alpha = 1/2 \pm A/\sqrt{n}$ and $\beta = 1/2 \pm B/\sqrt{n}$, where $A, B > 0$ are constants. Consider RLS-PD on $\text{Bilinear}^*{}_{\alpha,\beta}$. Let $\delta := \frac{1}{\sqrt{n}-1}$, $a := (A+B)\sqrt{n} + ((1+\delta)^2 - 1) \cdot (n/2 + (A+B)\sqrt{n})$ and $b := a + 1 + c\sqrt{n}\ln n$ for any $c > 0$. Then, for any $T > 1$ there exist constants $K > 0$ and $0 < \delta_0 < 1$ for which*

$$\Pr\left(\sum_{t=1}^{T} \mathbb{1}_{\{M_t \geq b\}} \geq 9T \cdot n^{1-c}\right) \leq K\delta_0^T.$$

Proof. We aim to use the occupation time bounds from Hajek [12, Theorem 3.1]. To do so we need to meet Conditions D1 and D2 and in addition show that the random variable M_0 is of exponential type. We begin with the latter.

The random variable M_0 is the sum of two random variables $X := \left||x_0|_1 - \beta n\right|$ and $Y := \left||y_0|_1 - \alpha n\right|$ where $|x_0|_1$ and $|y_0|_1$ denote the number of 1-bits in x_0 and y_0. Since both initial search points (x_0, y_0) are sampled uniformly at random from $\{0,1\}^n$, the number of 1-bits in (x_0, y_0) are sampled from a binomial distribution which by definition is of exponential type. Hence, M_0 is of exponential type.

Now, we show Condition D1: There exists $\eta > 0$ and $0 < \rho < 1$ for which $\mathrm{E}_t\left(e^{\eta(M_{t+1} - M_t)}; M_t > a\right) \leq \rho$. In the following we use $p^0_{x,y} = 1 - p^+_{x,y} - p^-_{x,y}$ to denote the probability that the Manhattan distance to the optimum does not change from step time t to $t+1$.

$$\begin{aligned}
&\mathrm{E}_t\left(e^{\eta(M_{t+1} - M_t)}; M_t > a\right) \\
&= p^0_{x,y} + p^+_{x,y}e^{\eta} + p^-_{x,y}e^{-\eta} \\
&= 1 - p^+_{x,y}(1 - e^{\eta}) - p^-_{x,y}(1 - e^{-\eta}) \\
&= 1 + p^+_{x,y}\delta - p^-_{x,y}\left(1 - \frac{1}{1+\delta}\right),
\end{aligned}$$

Where the last step uses $\eta := \ln(1+\delta) > 0$. By Lemma 4.4 $p^+_{x,y} \leq p^-_{x,y}/(1+\delta)^2$ for all $M_t > a$, hence,

$$\begin{aligned}
&\mathrm{E}_t\left(e^{\eta(M_{t+1} - M_t)}; M_t > a\right) \\
&\leq 1 + p^-_{x,y}\frac{\delta}{(1+\delta)^2} - p^-_{x,y}\frac{\delta}{1+\delta} \\
&= 1 - p^-_{x,y}\left(\frac{\delta}{1+\delta}\right)^2 \\
&\leq 1 - \frac{1}{8n},
\end{aligned}$$

where in the last step we used $p^+_{x,y} \geq 1/8$ (Lemma 4.4) and the definition $\delta := \frac{1}{\sqrt{n}-1}$.

In Condition D2 of the occupation time bounds, we require $\mathrm{E}_t\left(e^{\eta(M_{t+1} - a)}; M_t \leq a\right) \leq D$ with $D < \infty$. Since the Manhattan distance to the Nash equilibrium can only change by at most 1 and $M_t \leq a$ then $M_{t+1} - a \leq 1$. Hence, $\mathrm{E}_t\left(e^{\eta(M_{t+1} - a)}; M_t \leq a\right) \leq e^{\eta}$.

Now, the occupation time bounds tells us that

$$\Pr\left(\sum_{t=1}^{T} \mathbb{1}_{\{M_t < b\}} \leq T\rho_0(1-\varepsilon)\right) \leq K\delta_0^T,$$

for any $\varepsilon > 0$ and $\rho_0 = 1 - \frac{1}{1-\rho}De^{\eta(a-b)}$. This is equivalent to

$$\Pr\left(\sum_{t=1}^{T} \mathbb{1}_{\{M_t \geq b\}} > T(1 - \rho_0(1-\varepsilon))\right) \leq K\delta_0^T. \tag{7}$$

It remains to compute ρ_0.

$$\begin{aligned}
\rho_0 &= 1 - 8ne^{\eta(a-b+1)} \\
&= 1 - 8n\left(\frac{1}{1+\delta}\right)^{b-a-1}.
\end{aligned}$$

Using $\frac{1}{1+\delta} = 1 - \frac{\delta}{1+\delta} \leq \exp\left(-\frac{\delta}{1+\delta}\right) = \exp\left(\frac{1}{\sqrt{n}}\right)$ and the definition of $b := a + 1 + c\sqrt{n}\ln n$ yields $\rho_0 \geq 1 - 8n^{1-c}$. Plugging this back into Equation (7) and choosing $\varepsilon = \frac{n^{1-c}}{1-8n^{1-c}} > 0$ for some constant c proves the claim. □

Given the statement about occupation time [12], we can now derive an upper bound on the regret.

Theorem 4.6. *Let $\alpha = 1/2 \pm A/\sqrt{n}$ and $\beta = 1/2 \pm B/\sqrt{n}$, where $A, B > 0$ are constants. For any $T > 1$ the expected total regret of RLS-PD on $\text{Bilinear}^*{}_{\alpha,\beta}$ is $O(Tn^{1.5}\log n)$.*

Proof. Let $\mathcal{E}$ denote the event that $\sum_{t=1}^{T} \mathbb{1}_{\{M_t \geq b\}} < T\sigma$, for any $b > 0$ and $0 \leq \sigma \leq 1$. If the event is true the total regret R is at most $Tn\min\{\alpha, 1-\alpha\}((1-\sigma)b + \sigma\max_{0<t\leq T}(M_t))$ and $\max_{0<t\leq T}(M_t) \leq 2n$. Otherwise, the total regret is at most $Tn\min\{\alpha, 1-\alpha\}\max_{0<t\leq T}(M_t) \leq 2Tn^2\min\{\alpha, 1-\alpha\}$. Hence,

$$\begin{aligned}
\mathrm{E}(R) \leq \Pr(\mathcal{E}) \cdot Tn\min\{\alpha, 1-\alpha\}((1-\sigma)b + 2\sigma n) \\
+ \Pr\left(\overline{\mathcal{E}}\right) \cdot 2Tn^2\min\{\alpha, 1-\alpha\}.
\end{aligned}$$

Using $\sigma = n^{-2}$, Lemma 4.5 gives $\Pr\left(\overline{\mathcal{E}}\right) \leq K\delta_0^T$ for some constants $K > 0$ and $0 < \delta_0 < 1$. Hence,

$$\begin{aligned}
\mathrm{E}(R) \leq 1 \cdot Tn\min\{\alpha, 1-\alpha\}(b - bn^{-2} \\
+ 2n^{-1}) + K\delta_0^T \cdot 2Tn^2\min\{\alpha, 1-\alpha\}.
\end{aligned}$$

Note that $b = O(\sqrt{n}\log n)$ and $\min\{\alpha, 1-\alpha\} = O(1)$, therefore $\mathrm{E}(R) = O(Tn^{1.5}\log n)$. □

4.2 Improving RLS-PD with an Archive

In the previous sections we have seen that despite RLS-PD finding the Nash equilibrium efficiently, the algorithm forgets it and remains a large amount of time away from it. We show that a simple archive shown in Algorithm 2 solves this problem. Algorithm 2 shows RLS-PD embedded with the archive but the archive can be used in conjunction with any search based algorithm.

Algorithm 2 RLS-PD: Randomised Local Search with Pairwise Dominance with archive.

```
1:  Sample $x_1 \sim \mathrm{Unif}(\{0,1\}^n)$
2:  Sample $y_1 \sim \mathrm{Unif}(\{0,1\}^n)$
3:  Initialise archive $A_1 \leftarrow \{\}$
4:  for $t \in \{1,2,\dots\}$ do
5:      Create $x', y' \in \{0,1\}^n$ by copying $x_t$ and $y_t$ and flipping exactly one bit chosen uniformly at random from either $x_t$ or $y_t$.
6:      if $(x',y') \geq_g (x_t,y_t)$ then$(x_{t+1},y_{t+1}) := (x',y')$
7:      if $(x',y') \geq (x^*,y^*)\ \forall\ (x^*,y^*) \in A_t$ then
8:          $A' \leftarrow A_t \cup \{(x',y')\}$
9:      else
10:         $A' \leftarrow A_t$
11:     for $(x^*,y^*) \in A'$ do
12:         if $(x^*,y^*) \not\geq (x',y')$ then
13:             $A' \leftarrow A' \setminus \{(x^*,y^*)\}$
14:     $A_{t+1} \leftarrow A'$
```

In the following lemma we show that the archive maintains any Nash equilibria that it encounters solving the problem that RLS-PD has.

Lemma 4.7. *Let g be a* Maximin *function and $\tau \geq 1$. If $(x_\tau, y_\tau) \in$ OPT, then for every $t > \tau$ the archive A_t contains (x_τ, y_τ).*

Proof. By the definition of the set OPT neither the predator nor the prey benefit from changing strategy, that is, $g(x_\tau, y_\tau) \geq g(x, y_\tau)$ for all $x \in \mathcal{X}$ and $g(x_\tau, y_\tau) \leq g(x_\tau, y)$ for all $y \in \mathcal{Y}$. Then, by Definition 2.1 $(x_\tau, y_\tau) \geq_g (x, y)$ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$.

This means that at time $t = \tau$ the condition in Line 7 will be true because (x_τ, y_τ) dominates all other pairs of points and (x_τ, y_τ) will be added to the archive. Additionally, the condition in Line 12 will never be met for (x_τ, y_τ) because of the same reason. □

Using the archive for RLS-PD allows the algorithm to retain the Nash equilibrium, therefore, the regret of the algorithm can be improved significantly. In the following we consider the regret of the algorithm with respect to the best pair (x, y) in the archive at time t.

Theorem 4.8. *Let $\alpha = 1/2 \pm A/\sqrt{n}$ and $\beta = 1/2 \pm B/\sqrt{n}$, where $A, B > 0$ are constants and $3(A + B)^2 \leq 1/2 - \delta'$ for some constant $\delta' > 0$. Consider RLS-PD with archive on* Bilinear$^*_{\alpha,\beta}$*. Then, for any initial search points (x_0, y_0) and any $T \geq 1$, $\mathrm{E}(R) = O(n^{3.5})$.*

Proof. By Theorem 3.1 RLS-PD will reach a Nash equilibrium in expected $O(n^{1.5})$ iterations, each of these iterations contribute to the total regret by at most $2n^2 \min\{\alpha, 1 - \alpha\} = O(n^2)$. Therefore in expectation the total regret from the first iteration until a Nash equilibrium is found is $O(n^{3.5})$. Afterwards by Lemma 4.7 the archive will retain the Nash equilibrium, therefore all other iterations does not contribute to the total regret. □

We note that a simpler archive that retains all search points visited by the algorithm would obtain the same regret. But this archive would have $O(T)$ individuals. In the following lemma we show that the archive that we study is memory efficient.

Theorem 4.9. *Let $\alpha, \beta \in [0, 1]$. Let $\mathrm{OPT} = \{(x, y) \mid |x|_1 = \beta n \wedge |y|_1 = \alpha n\}$. Define $T := \inf\{t \mid (x_t, y_t) \in \mathrm{OPT}\}$, where (x_t, y_t) are the current solutions of RLS-PD. Consider RLS-PD with archive on* Bilinear$^*_{\alpha,\beta}$*. For all $t < T$, the archive size is at most 1 and for all $t > T$ the expected size of the archive is $O(|\mathrm{OPT}|)$.*

Proof. Let $t = 2$, then A_2 contains only (x_t, y_t). In order to add another pair to the archive the RLS-PD needs to create a pair of points that dominate (x_t, y_t). By Lemma 2.3 for this to happen the number of ones in either x or y must change by exactly one. Due to the nature of the Bilinear* function for two pairs of points $(x_1, y_1), (x_2, y_2) \notin \mathrm{OPT}$ with $(x_1, y_1) \geq (x_2, y_2)$, if $||x_1|_1 - |x_2|_1| = 1$ or $\big||y_1|_1 - |y_2|_1\big| = 1$ then $(x_2, y_2) \not\geq (x_1, y_1)$. Therefore, before reaching a Nash equilibrium, every time a pair of points are added to the archive, the previous pair of points of the archive are taken out, making the size of the archive at most 1.

Once a Nash equilibrium is added to the archive the only points that dominate it are Nash equilibria (therefore part of the set OPT) and points that have either $|x|_1 = \beta n \wedge |y|_1 = \alpha n \pm 1$ or $|x|_1 = \beta n \pm 1 \wedge |y|_1 = \alpha n$. Let t' denote a time step where $(x_{t'}, y_{t'})$ is a Nash equilibrium. Every time the optimum is reached there is a constant probability that in the next 8 iterations a point with exactly $M = 2$ in each quadrant is found. All other pairs in the archive that are not Nash equilibrium are dominated by a pairs with a Manhattan distance of 2 in the same quadrant as themselves. Therefore, after visiting a Nash equilibrium the algorithm will remove all pairs that are not Nash equilibrium with constant probability.

By Theorem 3.1 we reach a Nash equilibrium in $O(n^{1.5})$ iterations from any initial search point. And by the discussion above each time there is a constant probability to remove all pairs that are not Nash equilibrium. Therefore in expectation the size of the archive does not exceed $O(|\mathrm{OPT}| + n^{1.5})$. Noting that $n^{1.5} = o(|\mathrm{OPT}|)$ completes the proof. □

5 CONCLUSION

We proved that the RLS-PD identifies the Nash equilibrium for pseudo-Boolean bilinear zero-sum games, known as Bilinear*, in expected runtime $O(n^{1.5})$ and $\Omega(n)$. This is the first rigorous analysis of a (1+1)-type CoEA efficiently finding a zero-sum game's Nash equilibrium. Yet, RLS-PD tends to forget and quickly deviate from the found Nash equilibrium, leading to significant total regret. We derived a bound of $\tilde{\Theta}(Tn^{1.5})$ for this regret. Implementing a basic archive significantly improved RLS-PD's total regret.

To achieve our results, we developed a novel drift theorem that facilitates efficient expected hitting times for processes, even under minor negative drift. We believe this theorem holds standalone value and hope it will widen the runtime analysis toolbox for both EAs and CoEAs. Additionally, we proposed an innovative, intuitive method for integrating two separate potential functions that characterise the same underlying process.

Several open questions remain, for instance, how the considered algorithm would perform on other problems with more complex fitness landscapes and whether other (1+1)-type CoEAs will behave like RLS-PD, that is, it finds a Nash equilibrium efficiently but forgets it quickly and how about population-based CoEAs? It is also interesting to explore the impact of other methods to evaluate solutions (instead of pairwise dominance), mutation operators or crossover operators in these BILINEAR* problems.

ACKNOWLEDGMENTS

This research was supported by a Turing AI Fellowship (EPSRC grant ref EP/V025562/1).

REFERENCES

[1] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten. 2009. Min–max and min–max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research* 197, 2 (Sept. 2009), 427–438.

[2] Abdullah Al-Dujaili, Erik Hemberg, Una-May O'Reilly, and Tom Schmiedlechner. 2018. Towards Distributed Coevolutionary GANs. *CoRR* abs/1807.08194 (2018). arXiv:1807.08194 http://arxiv.org/abs/1807.08194

[3] Robert Axelrod et al. 1987. The evolution of strategies in the iterated prisoner's dilemma. *The dynamics of norms* 1 (1987), 1–16.

[4] Jürgen Branke and Johanna Rosenbusch. 2008. New approaches to coevolutionary worst-case optimization. In *Parallel Problem Solving from Nature - PPSN X (Lecture Notes in Computer Science, Vol. 5199)*. Springer, Dortmund, Germany, 144–153.

[5] Benjamin Doerr and Timo Kötzing. 2021. Multiplicative Up-Drift. *Algorithmica* 83, 10 (2021), 3017–3058.

[6] Benjamin Doerr and Frank Neumann (Eds.). 2020. *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization.* Springer International Publishing, Cham.

[7] Stefan Droste, Thomas Jansen, and Ingo Wegener. 2006. Upper and Lower Bounds for Randomized Search Heuristics in Black-Box Optimization. *Theory Comput. Syst.* 39, 4 (2006), 525–544.

[8] Diana Flores, Erik Hemberg, Jamal Toutouh, and Una-May O'Reily. 2022. Coevolutionary Generative Adversarial Networks for Medical Image Augumentation at Scale. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Boston, Massachusetts) *(GECCO '22)*. Association for Computing Machinery, New York, NY, USA, 367–376.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014.* Montreal, Quebec, Canada, 2672–2680.

[10] Sam Greenberg, Dana Randall, and Amanda Pascoe Streib. 2020. Sampling biased monotonic surfaces using exponential metrics. *Combinatorics, Probability and Computing* 29, 5 (2020), 672–697.

[11] Andreas Göbel, Timo Kötzing, and Martin S. Krejca. 2022. Intuitive Analyses via Drift Theory. arXiv:1806.01919 [math.PR]

[12] Bruce Hajek. 1982. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied probability* 14, 3 (1982), 502–525.

[13] Jun He and Xin Yao. 2001. Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence* 127, 1 (2001), 57–85.

[14] Mario Alejandro Hevia Fajardo and Per Kristian Lehre. 2023. How Fitness Aggregation Methods Affect the Performance of Competitive CoEAs on Bilinear Problems. In *Proceedings of the Genetic and Evolutionary Computation Conference.* ACM, to appear.

[15] W. Daniel Hillis. 1990. Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena* 42, 1-3 (June 1990), 228–234.

[16] Thomas Jansen and R. Paul Wiegand. 2004. The Cooperative Coevolutionary (1+1) EA. *Evolutionary Computation* 12, 4 (12 2004), 405–434.

[17] Mikkel T Jensen. 2003. A New Look at Solving Minimax Problems with Coevolution. *Metaheuristics: Computer Decision-Making* 86 (2003), 369.

[18] Timo Kötzing. 2016. Concentration of First Hitting Times Under Additive Drift. *Algorithmica* 75, 3 (July 2016), 490–506.

[19] Timo Kötzing and Martin S. Krejca. 2019. First-hitting times under drift. *Theoretical Computer Science* 796 (2019), 51–69.

[20] Timo Kötzing, Andrei Lissovoi, and Carsten Witt. 2015. (1+1) EA on Generalized Dynamic OneMax. In *Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII* (Aberystwyth, United Kingdom) *(FOGA '15)*. Association for Computing Machinery, New York, NY, USA, 40–51.

[21] Per Kristian Lehre. 2022. Runtime Analysis of Competitive Co-Evolutionary Algorithms for Maximin Optimisation of a Bilinear Function. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Boston, Massachusetts) *(GECCO '22)*. Association for Computing Machinery, New York, NY, USA, 1408–1416.

[22] P. K. Lehre and C. Witt. 2021. Tail bounds on hitting times of randomized search heuristics using variable drift analysis. *Combinatorics, Probability and Computing* 30, 4 (July 2021), 550–569. Publisher: Cambridge University Press.

[23] Tengyuan Liang and James Stokes. 2019. Interaction Matters: A Note on Non-asymptotic Local Convergence of Generative Adversarial Networks. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 907–915.

[24] Kristian Lindgren. 1991. Evolutionary phenomena in simple dynamics. *Artificial life II* 10 (1991), 295–312.

[25] Melanie Mitchell. 2006. Coevolutionary learning with spatially distributed populations. *Computational intelligence: principles and practice* 400 (2006).

[26] Michael Mitzenmacher and Eli Upfal. 2005. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis.* Cambridge University Press, New York, NY, USA.

[27] Aryan Mokhtari, Asuman Ozdaglar, and Sarath Pattathil. 2020. A Unified Analysis of Extra-gradient and Optimistic Gradient Methods for Saddle Point Problems: Proximal Point Approach. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 1497–1507.

[28] A. S. Nemirovski and D. B. Yudin. 1983. *Problem complexity and method efficiency in optimization.* Wiley.

[29] Elena Popovici, Anthony Bucci, R. Paul Wiegand, and Edwin D. De Jong. 2012. Coevolutionary Principles. In *Handbook of Natural Computing*, Grzegorz Rozenberg, Thomas Bäck, and Joost N. Kok (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 987–1033.

[30] Amirhossein Rajabi and Carsten Witt. 2020. Self-Adjusting Evolutionary Algorithms for Multimodal Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Cancún, Mexico) *(GECCO '20)*. ACM, New York, NY, USA, 1314–1322.

[31] Christopher Darrell Rosin. 1997. *Coevolutionary search among adversaries.* University of California, San Diego, California, USA.

[32] Jamal Toutouh, Erik Hemberg, and Una-May O'Reilly. 2019. Spatial evolutionary generative adversarial networks. In *Proceedings of the Genetic and Evolutionary Computation Conference.* ACM, Prague Czech Republic, 472–480.

[33] Rudolf Paul Wiegand. 2004. *An analysis of cooperative coevolutionary algorithms.* George Mason University, Virginia, USA.

[34] Nicholas C Wormald et al. 1999. The differential equation method for random graph processes and greedy algorithms. *Lectures on approximation and randomized algorithms* 73, 155 (1999), 0943–05073.

[35] Guojun Zhang and Yaoliang Yu. 2020. Convergence of Gradient Methods on Bilinear Zero-Sum Games. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020.*