Running Time Analysis of the (1+1)-EA using Surrogate Models on OneMax and LeadingOnes*

Zi-An Zhang, Chao Bian, and Chao Qian

State Key Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210023, China
{zhangza,bianc,qianc}@lamda.nju.edu.cn

Abstract Evolutionary algorithms (EAs) have been widely applied to solve real-world optimization problems. However, for problems where fitness evaluation is time-consuming, the efficiency of EAs is usually unsatisfactory. One common approach is to utilize surrogate models, which apply machine learning techniques to approximate the real fitness function. Though introducing noise, using surrogate models can reduce the time of fitness evaluation significantly, and has been shown usesful in many applications. However, the theretical analysis (especially the essential theoretical aspect, running time analysis) of surrogate-assisted EAs has not been studied. In this paper, we make a preliminary attempt by analyzing the running time of the (1+1)-EA using two typical kinds of preselection surrogate for solving the OneMax and LeadingOnes problems. The results imply that the running time can be significantly reduced when the surrogate model is accurate enough and properly used.

Keywords: Evolutionary algorithm · Surrogate model · Running time analysis.

1 Introduction

EAs, a kind of randomized heuristic optimization algorithm [2], have been widely used in real-world applications. However, the fitness (i.e., objective) evaluation for real-world problems is often very time-consuming. For example, in aerodynamic design [15], it is often necessary to carry out computational fluid dynamics simulations to evaluate the performance of a given structure, which is computationally expensive. Other examples include protein design, drug design, and material design [15]. The expensive fitness evaluation has limited the efficiency of EAs largely.

Much effort thus has been devoted to reducing the computational time in both the design and application of EAs. One popular idea is using machine learning models, called *surrogate models*, to approximate the real objective functions [15]. Specifically, it first samples some solutions from the solution space and

* This work was supported by the National Science Foundation of China (62022039). Chao Qian is the corresponding author.

evaluates their true fitness, and then uses them to train a learning model, which will be used to evaluate the newly generated solutions during the evolutionary process. Surrogate-assisted EAs have been widely used to solve real-world problems, e.g., the design of turbine blades, airfoils, forging, and vehicle crash tests [13]. Note that the idea of surrogate model also appears in other optimization methods, e.g., in Bayesian optimization where Gaussian processes are used as surrogate models [16,19].

However, it has been found that if only the surrogate model is used for fitness evaluation, EAs are very likely to converge to a false optimum [14]. Therefore, the surrogate model should be used together with the original fitness function in a proper way, leading to the issue of surrogate management [12,13,15]. Preselection is a widely used surrogate management strategy, which first expands the number of candidate offspring solutions and then uses the surrogate model to filter out some unpromising ones before the real fitness evaluation. Typical preselection mechanisms include the Regression model-based PreSelection (RPS) [10] which predicts the fitness of a solution, the Classification model-based PreSelection (CPS) [24] which predicts the probability of a solution being good, and the binary Relation Classification-based PreSelction (RCPS) [9] which predicts whether a solution is better than another one.

In contrast to the wide application of EAs, the theoretical foundation of EAs is underdeveloped due to their sophisticated behaviors. Much effort has been devoted to analyzing the essential theoretical aspect, i.e., running time complexity, of EAs [1,7,17,25]. The running time analysis starts from simple EAs solving synthetic problems. For example, one classical result is that the expected running time of the (1+1)-EA on OneMax and LeadingOnes is $\Theta(n \log n)$ and $\Theta(n^2)$, respectively [8]. Meanwhile, general running time analysis approaches, e.g., drift analysis [5,6,11,18], fitness-level methods [4,20,21], and switch analysis [3,22,23], have also been proposed. However, to the best of our knowledge, running time analysis of surrogate-assisted EAs has not been touched.

This paper aims at moving the first step towards running time analysis of surrogate-assisted EAs by considering the (1+1)-EA using the RPS and RCPS surrogates. Specifically, we first introduce a concept of (k, δ)-RPS surrogate, which generates k candidate offspring solutions in each iteration and predicts the fitness of a candidate solution wrong with probability δ, and then prove that the (1+1)-EA using the (k, δ)-RPS surrogate with $k = c/\delta$ (where c is a positive constant) and $\delta < 1/2$ can solve OneMax and LeadingOnes in $O(n + \delta n \log n)$ and $O(\max\{n, \delta n^2\})$ expected running time, respectively. The results show that the performance of EAs can be significantly improved, as long as δ is given appropriate values, e.g., $\delta = O(1/n)$. We also prove that the above upper bounds on the expected running time hold for the (1+1)-EA using the (k, δ)-RCPS surrogate, where δ denotes the probability of predicting the relation between any two offspring solutions wrong.

The rest of this paper starts with some preliminaries. Then, the running time analysis of the (1+1)-EA using the RPS and RCPS surrogates is presented in Sections 3 and 4, respectively. Section 5 concludes the paper.

2 Preliminaries

In this section, we first introduce EAs, surrogate models and problems studied in this paper, respectively, and then present the analysis tools that we use throughout this paper.

2.1 (1+1)-EA

The (1+1)-EA as described in Algorithm 1 is a simple EA for maximizing pseudo-Boolean functions over $\{0,1\}^n$. It reflects the common structure of EAs, and has been widely used in the running time analysis of EAs [1,17]. The (1+1)-EA maintains only one solution during the optimization procedure (i.e., the population size is 1), and repeatedly improves the current solution by using bit-wise mutation (i.e., line 3) and selection (i.e., lines 4–6).

Algorithm 1 (1+1)-EA

Given a function $f : \{0,1\}^n \to \mathbb{R}$ to be maximized:

1: $\boldsymbol{x}$:= uniformly randomly selected from $\{0,1\}^n$;
2: **repeat**
3: $\quad\boldsymbol{x}'$:= flip each bit of $\boldsymbol{x}$ independently with probability $1/n$;
4: $\quad$**if** $f(\boldsymbol{x}') \geq f(\boldsymbol{x})$ **then**
5: $\quad\quad\boldsymbol{x} := \boldsymbol{x}'$
6: $\quad$**end if**
7: **until** the termination condition is met

2.2 Surrogate Models

In this paper, we incorporate the widely used preselection surrogate model [9,10] into the (1+1)-EA. As described in Algorithm 2, the (1+1)-EA using preselection has the same general procedure as the original (1+1)-EA, i.e., it randomly generates an initial solution and improves it repeatedly. However, it inserts two key subprocedures: surrogate training (i.e., lines 1–2) and preselection (i.e., lines 5–8), which aim to train the surrogate model using the sampled data and use the surrogate model to select a promising solution, respectively. In the following, we present two specific preselection surrogates, i.e., RPS and RCPS surrogates, that will be studied in this paper.

RPS Surrogate tries to learn a mapping from the solution space to the objective space, based on a training set $P = \{\langle \boldsymbol{x}_i, f(\boldsymbol{x}_i)\rangle \,|i = 1, ..., N\}$, and then employs the mapping to predict the fitness value of newly generated candidate solutions. Note that $f(\boldsymbol{x})$ denotes the true fitness value of a solution. Specifically, we first sample a set of solutions from the solution space, and then employ a regression learning method, e.g., regression tree, to learn the mapping $\mathfrak{M}$. That is, line 2 of Algorithm 2 changes to

$$\mathfrak{M} = \text{RegressorTrain}(\{\langle \boldsymbol{x}_i, f(\boldsymbol{x}_i)\rangle \,|i = 1, ..., N\}).$$

Algorithm 2 (1+1)-EA with Preselection

1: Conduct a training data set P;
2: $\mathfrak{M}$ = SurrogateTrain(P);
3: $\boldsymbol{x}$:= uniformly randomly selected from $\{0,1\}^n$;
4: **repeat**
5: **for** $i = 1$ to k **do**
6: $\boldsymbol{u}_i$:= flip each bit of $\boldsymbol{x}$ independently with probability $1/n$
7: **end for**
8: $\boldsymbol{u}^*$ = PreSelection($\{\boldsymbol{u}_1, ..., \boldsymbol{u}_k\}, \mathfrak{M}$);
9: **if** $f(\boldsymbol{u}^*) \geq f(\boldsymbol{x})$ **then**
10: $\boldsymbol{x} := \boldsymbol{u}^*$
11: **end if**
12: **until** the termination condition is met

In the preselection procedure, we first generate k candidate offspring solutions, and then select a solution $\boldsymbol{u}^*$ which has the maximal predicted fitness value. That is, line 8 of Algorithm 2 changes to

$$\boldsymbol{u}^* = \arg\max_{\boldsymbol{u} \in \{\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_k\}} \text{Predict}(u, \mathfrak{M}),$$

where $\text{Predict}(\boldsymbol{u}, \mathfrak{M})$ denotes the fitness of the candidate solution $\boldsymbol{u}$ predicted by regression model $\mathfrak{M}$.

We introduce a concept of (k, δ)-RPS surrogate as presented in Definition 1, which will be used in our analysis. It specifies the number of solutions generated in lines 5–7 of Algorithm 2, as well as the accuracy of the surrogate model. That is, we omit the specific training methods, and only assume that the obtained preselection model can predict the fitness of a solution approximately correctly with some probability. Note that for the pseudo-Boolean functions considered in this paper, the acceptable threshold is set to 0.5, while for general problems, 0.5 can be replaced by a parameter ϵ.

Definition 1. *A (k, δ)-RPS surrogate is a regression model-based preselection surrogate such that*
(1) k offspring solutions are generated before the real fitness evaluation,
(2) the prediction error exceeds the acceptable threshold 0.5 with probability δ, i.e., $\mathrm{P}(|f(\boldsymbol{x}) - f'(\boldsymbol{x})| \geq 0.5) = \delta$, where $f'(\boldsymbol{x})$ denotes the predicted fitness of $\boldsymbol{x}$ by the surrogate.

RCPS Surrogate tries to learn a classifier which predicts whether a solution is better than another. Specifically, we first sample a set $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N\}$ of solutions from the solution space, and then employ Algorithm 3 to assign a label for each pair of solutions. That is, a pair $(\boldsymbol{x}, \boldsymbol{y})$ of solutions will be assigned a label 1 if $\boldsymbol{x}$ is better than $\boldsymbol{y}$ (i.e., $\boldsymbol{x}$ wins), and a label -1 otherwise. After that, we employ a classification learning method, e.g., decision tree, to learn the classifier $\mathfrak{M}$. That is, line 2 of Algorithm 2 changes to

$$\mathfrak{M} = \text{ClassifierTrain}(\{\langle(\boldsymbol{x}_i, \boldsymbol{x}_j), l\rangle \,|1 \leq i, j \leq N, i \neq j\}).$$

Algorithm 3 Training Data Preparation

```
for i = 1 to N do
  for j = 1 to i − 1 do
    if f(x_i) ≥ f(x_j) then
      assign the pair (x_i, x_j) a label l = 1
    else
      assign the pair (x_i, x_j) a label l = −1
    end if
    assign the pair (x_j, x_i) a label −l
  end for
end for
```

In the preselection procedure, we first generate k candidate offspring solutions, and then select a solution $\boldsymbol{u}^*$ which wins the most times in the pairwise competation, with ties broken uniformly. Note that for each pair of candidate offspring solutions, only one of them can win, i.e., $\forall i, j,\ \text{Predict}\left(\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right), \mathfrak{M}\right) = -\text{Predict}\left(\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right), \mathfrak{M}\right)$. That is, line 8 of Algorithm 2 changes to

$$\boldsymbol{u}^* = \arg\max_{\boldsymbol{u}\in\{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_k\}} \sum\nolimits_{\boldsymbol{u}_i\in\{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_k\}\setminus\{\boldsymbol{u}\}} \text{Predict}(\left(\boldsymbol{u}, \boldsymbol{u}_i\right), \mathfrak{M}),$$

where $\text{Predict}\left(\left(\boldsymbol{u}, \boldsymbol{u}_i\right), \mathfrak{M}\right)$ denotes the label of the pair $(\boldsymbol{u}, \boldsymbol{u}_i)$ of solutions predicted by classification model $\mathfrak{M}$.

Similar to the (k, δ)-RPS surrogate, in our analysis, we will omit the specific training methods, and only assume that the obtained classification model can predict the relation between two solutions correctly with some probability, as presented in Definition 2.

Definition 2. *A (k, δ)-RCPS surrogate is a binary relation classification-based preselction surrogate such that*
(1) k offspring solutions are generated before the real fitness evaluation,
(2) the relation between any two solutions is predicted wrong with probability δ.

2.3 OneMax and LeadingOnes

In this section, we introduce two well-known pseudo-Boolean functions OneMax and LeadingOnes, which will be used in this paper. The OneMax problem as presented in Definition 3 aims to maximize the number of 1-bits of a solution. Its optimal solution is 11...1 (briefly denoted as 1^n) with the function value n. It has been shown that the expected running time of the (1+1)-EA on OneMax is $\Theta(n \log n)$ [8]. For a Boolean solution $\boldsymbol{x}$, let x_i denote its i-th bit.

Definition 3 (Onemax). *The OneMax Problem of size n is to find an n bits binary string $\boldsymbol{x}^*$ such that $\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}\in\{0,1\}^n} \sum_{i=1}^{n} x_i$.*

The LeadingOnes problem as presented in Definition 4 aims to maximize the number of consecutive 1-bits counting from the left of a solution. Its optimal solution is 1^n with the function value n. It has been proved that the expected running time of the (1+1)-EA on LeadingOnes is $\Theta(n^2)$ [8].

Definition 4 (LeadingOnes). *The LeadingOnes Problem of size n is to find an n bits binary string $\boldsymbol{x}^*$ such that $\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}\in\{0,1\}^n} \sum_{i=1}^{n} \prod_{j=1}^{i} x_i$.*

2.4 Analysis Tools

Because an evolution process usually goes forward only based on the current population, an EA can be modeled as a Markov chain $\{\xi_t\}_{t=0}^{+\infty}$ [11,25]. The state space of the chain (denote as $\mathcal{X}$) is exactly the population space of the EA. The target state space $\mathcal{X}^*$ is the set of all optimal populations, where an "optimal" population implies containing an optimal solution. Note that we consider the discrete state space (i.e., $\mathcal{X}$ is discrete) in this paper.

Given a Markov chain $\{\xi_t\}_{t=0}^{+\infty}$ and $\xi_0 = \boldsymbol{x}$, we define its *first hitting time* (FHT) as a random variable τ such that $\tau = \min\{t|\xi_t \in \mathcal{X}^*, t \geq 0\}$. That is, τ is the number of generations required to reach the optimal state space $\mathcal{X}^*$ from $\xi_0 = \boldsymbol{x}$ for the first time. Then, we define the chain's *expected first hitting time* (EFHT) as the mathematical expectation of τ, i.e., $\mathbb{E}[\tau|\xi_0] = \sum_{i=0}^{+\infty} i \cdot \mathrm{P}(\tau = i)$.

In the following, we introduce two drift theorems which will be used to derive the EFHT of Markov chains in the paper. Drift analysis was first introduced to the running time analysis of EAs by He and Yao [11], and has become a popular tool with many variants [5,6]. We will use its additive (i.e., Lemma 1) as well as multiplicative (i.e., Lemma 2) version. To use drift analysis, we first need to construct a distance function $V(\boldsymbol{x})$ to measure the distance of a state x to the optimal state space $\mathcal{X}^*$, where $V(\boldsymbol{x})$ satisfies that $V(\boldsymbol{x} \in \mathcal{X}^*) = 0$ and $V(\boldsymbol{x} \notin \mathcal{X}^*) > 0$. Then, we need to investigate the progress on the distance to $\mathcal{X}^*$ in each step, i.e., $\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t]$. For additive drift analysis in Lemma 1, an upper bound on the EFHT can be derived through dividing the initial distance by a lower bound on the progress. Multiplicative drift analysis in Lemma 2 is much easier to use when the progress is roughly proportional to the current distance to the optimum.

Lemma 1 (Additive Drift [11]). *Given a Markov chain $\{\xi_t\}_{t=0}^{+\infty}$ and a distance function $V(x)$, if for any $t \geq 0$ and any ξ_t with $V(\xi_t) > 0$, there exists a real number $c > 0$ such that $\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t] \geq c$, then the EFHT satisfies that $\mathbb{E}[\tau|\xi_0] \leq V(\xi_0)/c$.*

Lemma 2 (Multiplicative Drift [6]). *Given a Markov chain $\{\xi_t\}_{t=0}^{+\infty}$ and a distance function $V(x)$, if for any $t \geq 0$ and any ξ_t with $V(\xi_t) > 0$, there exists a real number $c > 0$ such that $\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t] \geq c \cdot V(\xi_t)$, then the EFHT satisfies that $\mathbb{E}[\tau|\xi_0] \leq (1 + \ln(V(\xi_0)/V_{\min}))/c$, where $V_{\min}$ denotes the minimum among all possible positive values of V.*

3 Analysis of the (1+1)-EA using the RPS Surrogate

In this section, we analyze the expected running time of the (1+1)-EA using the (k, δ)-RPS surrogate on OneMax and LeadingOnes, respectively. Note that the acceptable threshold in Definition 1 is set to 0.5 on OneMax and LeadingOnes.

Under such setting, the condition (2) in Definition 1 implies that for any two solutions $\boldsymbol{x}$ and $\boldsymbol{y}$ with $f(\boldsymbol{x}) \geq f(\boldsymbol{y})$, $\boldsymbol{x}$ will be predicted better than $\boldsymbol{y}$ if the prediction error doesn't exceed the acceptable threshold.

We prove in Theorem 1 that when $\delta < 1/2$ and $k = c/\delta$, the expected running time of the (1+1)-EA using the (k, δ)-RPS surrogate on the OneMax problem is $O(n + \delta n \log n)$. Note that without surrogate model, the expected running time of the (1+1)-EA on the OneMax problem is $\Theta(n \log n)$ [8]. Therefore, if $\delta = O(1/\log n)$, the expected running time can be improved from $\Theta(n \log n)$ to $O(n)$. Intuitively, the results show that the running time can be significantly improved when the surrogate model is accurate enough and properly used.

The main proof idea can be summarized as follows. Since the comparison of the parent solution $\boldsymbol{x}$ and the preselected offspring solution $\boldsymbol{u}^*$ is under the real fitness, the distance function used in drift analysis does not increase. Furthermore, when at least one of the k offspring solutions is better than the parent, and all the k offspring solutions are "correctly" evaluated by the surrogate model, i.e., the prediction error doesn't exceed the acceptable threshold, there will be a positive progress on the distance function.

Theorem 1. *For the (1+1)-EA using the (k, δ)-RPS surrogate on the OneMax problem, the expected running time is $O(n + \delta n \log n)$ if $\delta < 1/2$ and $k = c/\delta$ (where c is a positive constant). Particularly, it is $O(n)$ if $\delta = O(1/\log n)$.*

Proof. We use additive and multiplicative drift analysis to prove this theorem. Let the distance function $V(\boldsymbol{x}) = |\boldsymbol{x}|_0$ be the number of 0-bits of a solution $\boldsymbol{x}$. It is easy to verify that $V(\boldsymbol{x} \in \mathcal{X}^* = \{1^n\}) = 0$ and $V(\boldsymbol{x} \notin \mathcal{X}^*) > 0$.

Suppose that the current solution $\boldsymbol{x}$ has i 0-bits, i.e., $|\boldsymbol{x}|_0 = i$. Then, we examine the expected progress $\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}]$. We decompose the progress into two parts, i.e., $\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] = E^+ - E^-$, where

$$E^+ = \sum\nolimits_{\xi_{t+1}:V(\xi_{t+1})<i} \mathrm{P}(\xi_{t+1}|\xi_t = \boldsymbol{x})(i - V(\xi_{t+1})),$$

$$E^- = \sum\nolimits_{\xi_{t+1}:V(\xi_{t+1})>i} \mathrm{P}(\xi_{t+1}|\xi_t = \boldsymbol{x})(V(\xi_{t+1}) - i).$$

That is, E^+ and E^- denote the positive and negative drift towards the optimal state, respectively. Since the comparison of the parent solution and the preselected offspring solution is under the real fitness, the fitness of the solution will never decrease. Thus, the distance function will not increase, implying $E^- = 0$. To analyze the positive drift E^+, we consider the probability that one offspring solution $\boldsymbol{x}'$ is better than the parent solution $\boldsymbol{x}$. We have

$$\mathrm{P}(f(\boldsymbol{x}') > f(\boldsymbol{x})) \geq (i/n) \cdot (1 - 1/n)^{n-1} \geq i/(en), \tag{1}$$

where the first inequality holds because it is sufficient to flip one of the i 0-bits of $\boldsymbol{x}$ by mutation and keep the other bits unchanged, and the second inequality is by $(1 - 1/n)^{n-1} \geq 1/e$. Then, we can derive a lower bound on the probability

of generating at least one offspring solution which is better than the parent solution, i.e.,

$$\begin{aligned}\mathrm{P}(\exists \boldsymbol{u}^* \in \{\boldsymbol{u}_j\}_{j=1}^k, f(\boldsymbol{u}^*) > f(\boldsymbol{x})) &\geq 1-(1-i/(en))^k \geq 1-e^{-ki/(en)} \\ &\geq 1-\frac{1}{1+ki/(en)} = ki/(ki+en),\end{aligned}$$

where the last two inequalities are both by $1+a \leq e^a$. When all the k offspring solutions are correctly evaluated by the surrogate model, whose probability is $(1-\delta)^k$, the best one will be chosen. Thus, we have

$$\mathrm{P}(V(\xi_{t+1}) < i | \xi_t = \boldsymbol{x}) \geq (ki/(ki+en)) \cdot (1-\delta)^k,$$

implying that

$$\begin{aligned}\mathbb{E}[V(\xi_t) - V(\xi_{t+1}) | \xi_t = \boldsymbol{x}] &\geq \mathrm{P}(V(\xi_{t+1}) < i | \xi_t = \boldsymbol{x}) \cdot 1 \\ &\geq (ki/(ki+en)) \cdot (1-\delta)^k.\end{aligned} \tag{2}$$

To derive the expected running time for finding the optimal solution, we divide the evolution process into two phases. The first phase starts from the initial solution and ends when $|\boldsymbol{x}|_0 \leq en/k$, and the second phase starts after the first phase finishes and ends when the optimal solution is found. Let τ_1 and τ_2 denote the running time of these two phases, respectively. For the first phase, i.e., $en/k \leq i \leq n$, because $ki/(ki+en) \geq ki/(ki+ki) = 1/2$, we have

$$\mathbb{E}[V(\xi_t) - V(\xi_{t+1}) | \xi_t = \boldsymbol{x}] \geq (1-\delta)^k/2.$$

Thus, by Lemma 1, we get

$$\mathbb{E}[\tau_1 | \xi_0] \leq 2n/(1-\delta)^k.$$

For the second phase, i.e., $i < en/k$, because $ki/(ki+en) \geq ki/(en+en) = ki/(2en)$, we have

$$\mathbb{E}[V(\xi_t) - V(\xi_{t+1}) | \xi_t = \boldsymbol{x}] \geq ki\,(1-\delta)^k/(2en),$$

Thus, by Lemma 2, we get

$$\mathbb{E}[\tau_2 | \xi_0] \leq \frac{1+\ln\frac{en}{k}}{\frac{k(1-\delta)^k}{2en}} = \frac{2e\,(2-\ln k)}{k\,(1-\delta)^k} n + \frac{2e}{k\,(1-\delta)^k} n \ln n.$$

Combining the analysis of the two phases, we have

$$\mathbb{E}\left[\tau | \xi_0\right] = \mathbb{E}\left[\tau_1 | \xi_0\right] + \mathbb{E}\left[\tau_2 | \xi_0\right] \leq \frac{1}{\left((1-\delta)^{\frac{1}{\delta}}\right)^c} \cdot \left(2n + \frac{4e}{c}\delta n + \frac{2e}{c}\delta n \ln n\right),$$

where the last inequality is by $k = c/\delta$. Note that

$$\begin{aligned}\frac{1}{\left((1-\delta)^{\frac{1}{\delta}}\right)^c} &= \left(\left(1+\frac{\delta}{1-\delta}\right)^{\frac{1}{\delta}-1}\right)^c \cdot \frac{1}{(1-\delta)^c} \\ &\leq e^{\frac{\delta}{1-\delta}\cdot(\frac{1}{\delta}-1)\cdot c} \cdot \frac{1}{(1-\delta)^c} = \left(\frac{e}{1-\delta}\right)^c < (2e)^c,\end{aligned} \tag{3}$$

where the first inequality is by $1 + a \leq e^a$, and the second inequality is by $\delta < 1/2$. Furthermore, as c is a constant, we get $\mathbb{E}[\tau|\xi_0] = O(n + \delta n \log n)$. Thus, the theorem holds. □

We prove in Theorem 2 that when $\delta < 1/2$ and $k = c/\delta$, the expected running time of the (1+1)-EA using the (k, δ)-RPS surrogate on the LeadingOnes problem is $O(\max\{n, \delta n^2\})$. Note that without surrogate model, the expected running time of the (1+1)-EA on the LeadingOnes problem is $\Theta(n^2)$ [8]. Therefore, if $\delta = O(1/n)$, the expected running time can be improved from $\Theta(n^2)$ to $O(n)$. The main proof idea is similar to that of Theorem 1. That is, the distance function used in drift analysis does not increase; meanwhile, it can decrease if at least one of the offspring solutions is better than the parent solution and all the offspring solutions are correctly evaluated by the surrogate model.

Theorem 2. *For the (1+1)-EA using the (k, δ)-RPS surrogate on the LeadingOnes problem, the expected running time is $O(\max\{n, \delta n^2\})$ if $\delta < 1/2$ and $k = c/\delta$ (where c is a positive constant). Particularly, it is $O(n)$ if $\delta = O(1/n)$.*

Proof. We use additive drift analysis to prove this theorem. Let the distance function $V(\boldsymbol{x}) = n - LO(\boldsymbol{x})$, where $LO(\boldsymbol{x})$ is the number of leading 1-bits of $\boldsymbol{x}$. It is easy to verify that $V(\boldsymbol{x} \in \mathcal{X}^* = \{1^n\}) = 0$ and $V(\boldsymbol{x} \notin \mathcal{X}^*) > 0$. Suppose that the current solution $\boldsymbol{x}$ has i leading 1-bits, i.e., $LO(\boldsymbol{x}) = i < n$. Then, Eq. (1) becomes

$$\mathrm{P}(f(\boldsymbol{x}') > f(\boldsymbol{x})) \geq (1/n) \cdot (1-1/n)^i \geq 1/(en), \tag{4}$$

since it is sufficient to flip the first 0-bit and keep the i leading 1-bits unchanged. Eq. (2) becomes

$$\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] \geq \mathrm{P}(V(\xi_{t+1}) < i|\xi_t = \boldsymbol{x}) \cdot 1 \geq (k/(k+en)) \cdot (1-\delta)^k.$$

We consider two cases for k. If $k \geq en$, we have $k/(k + en) \geq 1/2$. Then, we get $\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] \geq (1-\delta)^k/2$. By Lemma 1, we get

$$\mathbb{E}[\tau|\xi_0] \leq \frac{2n}{(1-\delta)^k} = \frac{1}{\left((1-\delta)^{\frac{1}{\delta}}\right)^c} \cdot 2n = O(n),$$

where the first equality is by $k = c/\delta$, and the second inequality holds by Eq. (3). If $k < en$, we have $k/(k + en) \geq k/(2en)$. Then, we get $\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] \geq k(1-\delta)^k/(2en)$. By Lemma 1, we get

$$\mathbb{E}[\tau|\xi_0] \leq \frac{2en^2}{k\,(1-\delta)^k} = O(\delta n^2),$$

Thus, the analysis of the above two cases leads to $\mathbb{E}[\tau|\xi_0] = O\left(\max\{n, \delta n^2\}\right)$, implying that the theorem holds. □

4 Analysis of the (1+1)-EA using the RCPS Surrogate

In this section, we analyze the expected running time of the (1+1)-EA using the (k, δ)-RCPS surrogate on OneMax and LeadingOnes, respectively.

We prove in Theorem 3 that when $\delta < 1/2$ and $k = c/\delta$, the expected running time of the (1+1)-EA using the (k, δ) RCPS surrogate on the OneMax problem is $O(n + \delta n \log n)$. Thus, the expected running time can be reduced by a factor of $O(\log n)$ if $\delta = O(1/\log n)$, which also suggests the effectiveness of the surrogate model. The main proof idea can be summarized as follows. Similar to the proof of Theorem 1, the distance function does not increase. When some offspring solutions are better than the parent solution and one of these offspring solutions wins the competition with the other offspring solutions, the preselected offspring solution can be better than the parent solution, leading to a positive drift towards the optimal solution.

Theorem 3. *For the (1+1)-EA using the (k, δ)-RCPS surrogate on the OneMax problem, the expected running time is $O(n + \delta n \log n)$ if $\delta < 1/2$ and $k = c/\delta$ (where c is a positive constant). Particularly, it is $O(n)$ if $\delta = O(1/\log n)$.*

Proof. We use additive and multiplicative drift analysis to prove this theorem. Let the distance function $V(\boldsymbol{x}) = |\boldsymbol{x}|_0$ be the number of 0-bits of a solution $\boldsymbol{x}$. Suppose that the current solution $\boldsymbol{x}$ has i 0-bits, i.e., $|\boldsymbol{x}|_0 = i$.

Suppose that the offspring solutions $\boldsymbol{x}_1, ..., \boldsymbol{x}_m$ are better the parent solution $\boldsymbol{x}$ and $\boldsymbol{x}_{m+1}, ..., \boldsymbol{x}_k$ are worse than the parent solution (or have the same fitness as the parent solution). If $\exists j \in \{1, ..., m\}$, $\boldsymbol{x}_j$ wins the competitions with all the other offspring solutions, then $\boldsymbol{x}_j$ will be chosen and will bring progress as it is better than the parent solution. The probability of this event is

$$\sum_{j=1}^{m} \delta^{j-1}(1-\delta)^{k-j} = \frac{(1-\delta)^k}{1-2\delta}\left(1 - \left(\frac{\delta}{1-\delta}\right)^m\right).$$

Let p denote the probability that a better individual is produced by mutation, which is at least $i/(en)$ by Eq. (1). Then, we have

$$\begin{aligned}
\mathrm{P}(V(\xi_{t+1}) < i | \xi_t = \boldsymbol{x}) &\geq \sum_{m=1}^{k} \binom{k}{m} p^m (1-p)^{k-m} \frac{(1-\delta)^k}{1-2\delta}\left(1 - \left(\frac{\delta}{1-\delta}\right)^m\right) \\
&= \frac{(1-\delta)^k}{1-2\delta}\left(1 - \left(1 - \frac{1-2\delta}{1-\delta}p\right)^k\right) \\
&\geq \frac{(1-\delta)^k}{1-2\delta}\left(1 - \frac{1}{1 + \frac{1-2\delta}{1-\delta}\frac{ki}{en}}\right).
\end{aligned}$$

Similar to the analysis of Theorem 1, we have

$$\begin{aligned}\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] &\geq \mathrm{P}(V(\xi_{t+1}) < i|\xi_t = \boldsymbol{x}) \cdot 1 \\ &\geq \frac{(1-\delta)^k}{1-2\delta}\left(1 - \frac{1}{1+\frac{1-2\delta}{1-\delta}\frac{ki}{en}}\right).\end{aligned}$$

To derive the expected running time for finding the optimal solution, we divide the evolutionary process into two phases. The first phase starts from the initial solution and ends when $|\boldsymbol{x}|_0 \leq \frac{1-\delta}{1-2\delta}\frac{en}{k}$, and the second phase starts after the first phase finishes and ends when the optimal solution is found. Let τ_1 and τ_2 denote the running time of these two phases, respectively. For the first phase, i.e., $\frac{1-\delta}{1-2\delta}\frac{en}{k} \leq i \leq n$, we have

$$\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] \geq (1-\delta)^k/(2-4\delta),$$

By Lemma 1, we get

$$\mathbb{E}[\tau_1|\xi_0] \leq n(2-4\delta)/(1-\delta)^k \leq 2n/(1-\delta)^k.$$

For the second phase, i.e., $i < \frac{1-\delta}{1-2\delta}\frac{en}{k} \leq n$, we have

$$\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] \geq ki(1-\delta)^{k-1}/(2en).$$

By Lemma 2, we get

$$\mathbb{E}[\tau_2|\xi_0] \leq \frac{1+\ln\left(\frac{1-\delta}{1-2\delta}\frac{en}{k}\right)}{\frac{k(1-\delta)^{k-1}}{2en}} \leq \frac{2en(1+\ln n)}{k\,(1-\delta)^k}.$$

Combining the analysis of the two cases, we have

$$\mathbb{E}\left[\tau|\xi_0\right] = \mathbb{E}\left[\tau_1|\xi_0\right] + \mathbb{E}\left[\tau_2|\xi_0\right] \leq \frac{2n}{(1-\delta)^k} + \frac{2en(1+\ln n)}{k(1-\delta)^k} = O(n + \delta n \log n),$$

where the last equality is by $k = c/\delta$ and Eq. (3). Thus, the theorem holds. □

We prove in Theorem 4 that when $\delta < 1/2$ and $k = c/\delta$, the expected running time of the (1+1)-EA using the (k,δ)-RCPS surrogate on the LeadingOnes problem is $O(\max\{n, \delta n^2\})$. The results show that the expected running time can be reduced by a factor of $O(n)$ if $\delta = O(1/n)$. The main proof idea is similar to that of Theorem 3.

Theorem 4. *For the (1+1)-EA using the (k,δ)-RCPS surrogate on the LeadingOnes problem, the expected running time is $O(\max\{n, \delta n^2\})$ if $\delta < 1/2$ and $k = c/\delta$ (where c is a positive constant). Particularly, it is $O(n)$ if $\delta = O(1/n)$.*

Proof. We use additive drift analysis to prove this theorem. Let the distance function $V(\boldsymbol{x}) = n - LO(\boldsymbol{x})$. Suppose that the current solution $\boldsymbol{x}$ has i leading 1-bits, i.e., $LO(\boldsymbol{x}) = i < n$.

Similar to the analysis in Theorem 3, we have

$$\begin{aligned}\mathrm{P}(V(\xi_{t+1}) < i|\xi_t = \boldsymbol{x}) &\geq \frac{(1-\delta)^k}{1-2\delta}\left(1 - \frac{1}{1+\frac{1-2\delta}{1-\delta}pk}\right) \\ &\geq \frac{(1-\delta)^{\frac{c}{\delta}}}{1-2\delta}\left(1 - \frac{1}{1+\left(\frac{1}{\delta}-2\right)\frac{c}{en}}\right),\end{aligned}$$

where the second inequality holds by $p \geq 1/(en)$ as shown in Eq. (4), $k = c/\delta$, and $\delta < 1/2$. We consider two cases for δ. If $\left(\frac{1}{\delta}-2\right)\cdot\frac{c}{en} \geq 1$, i.e., $\delta \leq \frac{c}{en+2c}$, we have

$$\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] \geq \mathrm{P}(V(\xi_{t+1}) < i|\xi_t = \boldsymbol{x}) \cdot 1 \geq (1-\delta)^{c/\delta}/2.$$

By Lemma 1, we get

$$\mathbb{E}[\tau|\xi_0] \leq 2n/(1-\delta)^{c/\delta} = O(n).$$

If $\left(\frac{1}{\delta}-2\right)\cdot\frac{c}{en} < 1$, i.e., $\delta > \frac{c}{en+2c}$, we have

$$\mathbb{E}[V(\xi_t) - V(\xi_{t+1})|\xi_t = \boldsymbol{x}] \geq c\,(1-\delta)^{c/\delta}/(2e\delta n).$$

By Lemma 1, we get

$$\mathbb{E}[\tau|\xi_0] \leq \frac{2e\delta n^2}{c\,(1-\delta)^{c/\delta}} = O(\delta n^2).$$

Thus, $\mathbb{E}[\tau|\xi_0] = O\left(\max\{n, \delta n^2\}\right)$, implying that the theorem holds. □

5 Conclusion and Discussion

In this paper, we conduct a preliminary study on the running time analysis of surrogate-assisted EAs, by considering the (1+1)-EA using the RPS and RCPS surrogates solving OneMax and LeadingOnes. We introduce the concept of the (k, δ)-RPS and (k, δ)-RCPS surrogates, and derive the parameter values that can make using the surrogate model accelerate the evolution process. The results imply that if the surrogate model is accurate enough and used properly, the running time can be significantly improved.

We hope this work can encourage more work on the running time analysis of EAs using surrogate models. In the future, the following two directions can be considered. On one hand, in this paper, we simply assume that the surrogate model is trained in advance before optimization, while in practical applications, the surrogate model is usually updated along with the optimization process. It is interesting to theoretically study the impact of updating the surrogate model with newly obtained data. On the other hand, when analyzing the running time, we only consider the cost of true fitness evaluation during the evolutionary process, which is somewhat unfair. It is interesting to examine the total cost of surrogate-assisted EAs, i.e., the cost of training before evolution and the cost during the evolutionary process.

References

1. Auger, A., Doerr, B.: Theory of Randomized Search Heuristics: Foundations and Recent Developments. World Scientific, Singapore (2011)
2. Back, T.: Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford University Press, Oxford, UK (1996)
3. Bian, C., Qian, C., Tang, K.: A general approach to running time analysis of multi-objective evolutionary algorithms. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18). pp. 1405–1411. Stockholm, Sweden (2018)
4. Corus, D., Dang, D.C., Eremeev, A.V., Lehre, P.K.: Level-based analysis of genetic algorithms and other search processes. IEEE Transactions on Evolutionary Computation **22**(5), 707–719 (2017)
5. Doerr, B., Goldberg, L.A.: Adaptive drift analysis. Algorithmica **65**(1), 224–250 (2013)
6. Doerr, B., Johannsen, D., Winzen, C.: Multiplicative drift analysis. Algorithmica **64**(4), 673–697 (2012)
7. Doerr, B., Neumann, F.: Theory of Evolutionary Computation: Recent Developments in Discrete Optimization. Springer, Cham, Switzerland (2020)
8. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science **276**(1-2), 51–81 (2002)
9. Hao, H., Zhang, J., Lu, X., Zhou, A.: Binary relation learning and classifying for preselection in evolutionary algorithms. IEEE Transactions on Evolutionary Computation **24**(6), 1125–1139 (2020)
10. Hao, H., Zhang, J., Zhou, A.: A comparison study of surrogate model based preselection in evolutionary optimization. In: Proceedings of the 14th International Conference on Intelligent Computing Theories and Application (ICIC). pp. 717–728. Wuhan, China (2018)
11. He, J., Yao, X.: Drift analysis and average time complexity of evolutionary algorithms. Artificial intelligence **127**(1), 57–85 (2001)
12. Jin, Y.: A comprehensive survey of fitness approximation in evolutionary computation. Soft Computing **9**(1), 3–12 (2005)
13. Jin, Y.: Surrogate-assisted evolutionary computation: Recent advances and future challenges. Swarm and Evolutionary Computation **1**(2), 61–70 (2011)
14. Jin, Y., Olhofer, M., Sendhoff, B.: A framework for evolutionary optimization with approximate fitness functions. IEEE Transactions on Evolutionary Computation **6**(5), 481–494 (2002)
15. Jin, Y., Wang, H., Sun, C.: Data-Driven Evolutionary Optimization. Springer, Cham, Switzerland (2021)
16. Mockus, J.: Application of Bayesian approach to numerical methods of global and stochastic optimization. Journal of Global Optimization **4**(4), 347–365 (1994)
17. Neumann, F., Witt, C.: Bioinspired Computation in Combinatorial Optimization: Algorithms and Their Computational Complexity. Springer, Berlin, Germany (2010)
18. Oliveto, P.S., Witt, C.: Simplified drift analysis for proving lower bounds in evolutionary computation. Algorithmica **59**(3), 369–386 (2011)
19. Qian, C., Xiong, H., Xue, K.: Bayesian optimization using pseudo-points. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI'20). pp. 3044–3050. Yokohama, Japan (2020)

20. Sudholt, D.: A new method for lower bounds on the running time of evolutionary algorithms. IEEE Transactions on Evolutionary Computation **17**(3), 418–435 (2012)
21. Wegener, I.: Methods for the analysis of evolutionary algorithms on pseudo-Boolean functions. In: Evolutionary Optimization, pp. 349–369. Kluwer, Norwell, MA (2002)
22. Yu, Y., Qian, C.: Running time analysis: Convergence-based analysis reduces to switch analysis. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC). pp. 2603–2610. Sendai, Japan (2015)
23. Yu, Y., Qian, C., Zhou, Z.H.: Switch analysis for running time analysis of evolutionary algorithms. IEEE Transactions on Evolutionary Computation **19**(6), 777–792 (2014)
24. Zhang, J., Zhou, A., Tang, K., Zhang, G.: Preselection via classification: A case study on evolutionary multiobjective optimization. Information Sciences **465**, 388–403 (2018)
25. Zhou, Z.H., Yu, Y., Qian, C.: Evolutionary Learning: Advances in Theories and Algorithms. Springer, Singapore (2019)