On Sampling-and-Classification Optimization in Discrete Domains

Hong Qian
National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210023, China
Email: qianh@lamda.nju.edu.cn

Yang Yu
National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210023, China
Email: yuy@lamda.nju.edu.cn

***Abstract*—As a kind of model-based optimization framework, the sampling-and-classification (SAC) algorithms, where the model is specified to be a classifier, has been recently studied in both theoretical foundation and algorithm implementation. However, the previous work only studied SAC algorithms in real domains. While significant progresses of theoretical evolutionary algorithm have been developed major in discrete domains, it is interesting to understand the SAC algorithms also in finite discrete domains.**

This paper studies the (ϵ, δ)-query complexity of SAC algorithms, which measures how soon can an algorithm obtain a solution with the desired approximation quality with a probability. Some classical pseudo-Boolean problems are employed to probe the SAC algorithms, including OneMax problem, linear pseudo-Boolean functions, LeadingOnes problem, and Trap problem. The theoretical results disclose that SAC algorithms can achieve a small complexity for approximating these problems. Moreover, an implementation of the SAC framework, the RACOS algorithm, is compared with the well-analyzed (1+1)-EA on these problems.

I. Introduction

Difficult optimization tasks are often encountered in numerous real-world applications and are at the core place in many fields. Evolutionary algorithms (EAs) are perhaps the largest family of heuristic optimization algorithms, covering genetic algorithms, evolutionary programming, evolutionary strategies, particle swarm optimization, ant colony optimization, estimation of distribution algorithms, etc. Evolutionary algorithms, which belong to black-box search algorithms, are direct search algorithms and can be applied to tackle the optimization tasks. They have been reported to achieve many successful applications. Besides the showcases, it is equally important to theoretically understand this kind of algorithms, in order to further improve their performance. In recent decades, theoretical studies of evolutionary algorithms have made a significant leap, including the development of analyzing techniques (e.g., [1], [2], [3], [4]), the analyzed problems (e.g., [5], [6]), the algorithm components (e.g., [7], [8], [9], [10], [11], [12]), the performance evaluations (e.g., [13], [6], [14], [15], [16], [17]). In addition, some progresses have been aggregated in recent books [18], [19], [20].

This research was supported by the NSFC (61375061, 61223003), Foundation for the Author of National Excellent Doctoral Dissertation of China (201451), and 2015 Microsoft Research Asia Collaborative Research Program.

Some model-based optimization methods, such as estimation of distribution algorithms [21] and the cross-entropy method [22], can be abstracted as a sampling-and-learning framework [23], [24], which mainly consists of the cycle of a sampling step and a learning step. In the sampling step, it generates solutions from a model (or a distribution), which is initially an uniform distribution over the solution space. In the learning step, it learns a model from the solutions together with their fitness values. For the ease of analysis, the case where the learning model is a binary classification model has been studied [24], which forms the sampling-and-classification (SAC) algorithms. In machine learning, classification is a subtype of learning that classifies a sample into one of the candidate classes. In SAC algorithms, the binary classification is used to classify the solutions into two classes, i.e., the good (or positive) and the bad (or negative). The resulting classification model splits the whole solution space completely into two disjoint regions, one for the good solutions and the other for the bad solutions. Then in the sampling step, a SAC algorithm samples from the region of the good solutions.

The (ϵ, δ)-query complexity discloses the number of fitness evaluations, which is considered as the core complexity of evolutionary algorithms, for achieving solutions with ϵ additive approximation quality with probability $1-\delta$. The (ϵ, δ)-query complexity can reflect closely what evolutionary algorithms are expected in practice: obtaining a satisfactory solution, but may not be an optimal one, after multiple trials. In [24], the performance of SAC algorithms on Sphere function and the Spike function was derived. In [25], two critical factors related to the performance of SAC algorithms were identified, and following that, a SAC algorithm named RACOS was designed. However, previous studies mainly focused on studying SAC algorithms in real domains, it is also interesting to understand their performance in finite discrete domains.

In this paper, we conduct theoretical analysis of the classification-based optimization method, a kind of simplified SAC algorithms, on some classical discrete problems, including OneMax problem, linear pseudo-Boolean functions, LeadingOnes problem, and Trap problem. The theoretical results show that the classification-based optimization method can be efficient approximation optimizers, as their (ϵ, δ)-query complexities are all in the order of $O(n \log n)$ ignoring the ϵ and δ. We also empirically studied the RACOS algorithm by comparing with (1+1)-EA on OneMax and LeadingOnes problems, which shows that the implementation of such method can also be efficient.

The rest of this paper is organized in 4 sections, sequentially presenting the preliminaries, theoretical analysis results, experimental results, and conclusion.

II. Preliminaries

A. Problems and Performance Evaluation

In this paper, we always consider general minimization problems in finite discrete domains. Let X denote the finite discrete solution space which is a subset of $\mathbb{R}^n$, and $f\colon X \to \mathbb{R}$ denote a minimization problem. It will not affect the generality that the investigated problems are restricted to minimization problems throughout the paper, since all maximization problems can be mapped to minimization problems by multiplying the objective (fitness) function by -1. Since $X \subseteq \mathbb{R}^n$ is finite, there always exist $x^*, x' \in X$ such that $f(x^*) = \min_{x\in X} f(x), f(x') = \max_{x\in X} f(x)$. In a nutshell, the investigated problems in this paper can be summarized as Definition 1.

Definition 1 (Minimization Problem) *A minimization problem consists of a finite discrete solution space $X \subseteq R^n$ and a function $f\colon X \to \mathbb{R}$. The goal is to find a solution $x^* \in X$ such that $f(x^*) \le f(x)$ for all $x \in X$.*

For evaluating the optimization performance, we will apply the (ϵ, δ)-query complexity (Definition 2) [24], [25]. The (ϵ, δ)-query complexity counts the total number of calls to the objective function by an algorithm before it finds a solution that reaches the additive approximation level ϵ, with probability at least $1 - \delta$. However, for discrete domains, there may not exist a corresponding solution at arbitrary approximation level. Therefore, we assume that ϵ can only be chosen from the finite set $E = \{f(x) - f(x^*) \mid \forall x \in X\}$, where $0 \in E$.

Definition 2 ((ϵ, δ)-Query Complexity) *Given a minimization problem f, an algorithm $\mathcal{A}$, $0 < \delta < 1$ and $\epsilon > 0$, the (ϵ, δ)-query complexity is the number of calls to f such that, with probability at least $1 - \delta$, $\mathcal{A}$ finds at least one solution $\tilde{x} \in X \subseteq \mathbb{R}^n$ satisfying*

$$f(\tilde{x}) - f(x^*) \le \epsilon,$$

where $f(x^*) = \min_{x\in X} f(x)$.

The (ϵ, δ)-query complexity closely reflects our intuitive evaluation of EAs in real-world practice, where we expect EAs to achieve some good enough solutions with a sufficient large probability. Definition 2 indicates that the (ϵ, δ)-query complexity characterizes the optimization performance of an algorithm via the approximation level parameter ϵ, the confidence parameter δ, and the solution space dimension parameter n. Let $\mathtt{poly}(\cdot)$ denote the set of all polynomials w.r.t. the related variables, and $\mathtt{superpoly}(\cdot)$ denote the set of all functions that grow faster than any function in $\mathtt{poly}(\cdot)$ w.r.t. the related variables. Given a minimization problem f, if the (ϵ, δ)-query complexity of an algorithm $\mathcal{A}$ belongs to $\mathtt{poly}(\frac{1}{\epsilon}, \frac{1}{\delta}, n)$, we say that f can be efficiently approximated by $\mathcal{A}$.

B. Classification-Based Optimization Method

Among a diverse range of methods that can handle the minimization problems formalized in Definition 1, a classification-based optimization method [25], which is inspired from the statistical view of EAs, is recently proposed. The classification-based optimization method is theoretically grounded and has a desirable performance for a large range of problems. This method is a simplified version of the sampling-and-classification (SAC) framework proposed in [24].

As most of EAs, the classification-based optimization method assumes that only an oracle of the objective function is accessible, i.e., given any solution $x \in X$, an algorithm can query the oracle to obtain $f(x)$. In the paper, we assume that, given f, lower and upper bounds of f are known. Let $\underline{M}_f$ and $\overline{M}_f$ be its lower and upper bounds, respectively. For a subset $D \subseteq X$, let $\#D = \sum_{x\in X} \mathbb{I}[x \in D]$, where $\mathbb{I}[\cdot]$ is the indicator function. Define $|D| = \#D/\#X$ and thus $|D| \in [0, 1]$. Let $D_\alpha = \{x \in X \mid f(x) \le \alpha\}$ for $\alpha \in [\underline{M}_f, \overline{M}_f]$, and $D_\epsilon = \{x \in X \mid f(x) - f(x^*) \le \epsilon\}$ for $\epsilon > 0$.

A hypothesis (or a classifier) h is a function mapping the solution space X to $\{-1, +1\}$. Let $\mathcal{H} \subseteq \{h\colon X \to \{-1, +1\}\}$ be a hypothesis space consisting of candidate hypotheses h. Let $D_h = \{x \in X \mid h(x) = +1\}$ for any hypothesis $h \in \mathcal{H}$, i.e., the positive class region represented by h. Denote $\mathcal{U}_X$ and $\mathcal{U}_{D_h}$ the uniform distribution over X and D_h, respectively, and denote $\mathcal{T}_h$ the distribution defined on D_h induced by h. Let $\mathrm{sign}[v]$ be the sign function that returns 1 if $v \ge 0$ and -1 otherwise.

Algorithm 1 Classification-Based Optimization

```
Input:
    $f$: Objective function to be minimized;
    $\mathcal{C}$: A binary classification algorithm;
    $\mathcal{H}$: Hypothesis space;
    $\lambda \in (0, 1)$: Balancing parameter;
    $\alpha_1 > \ldots > \alpha_T$: Threshold for labeling;
    $T \in \mathbb{N}^+$: Number of iterations;
    $m \in \mathbb{N}^+$: Sample size in each iteration.
Procedure:
 1: Collect $S_0 = \{x_1, \ldots, x_m\}$ by i.i.d. sampling from $\mathcal{U}_X$
 2: Let $\tilde{x} = \mathrm{argmin}_{x\in S_0} f(x)$
 3: for $t = 1$ to $T$ do
 4:     Construct $B_t = \{(x_1, y_1), \ldots, (x_m, y_m)\}$,
        where $x_i \in S_{t-1}$ and $y_i = \mathrm{sign}[\alpha_t - f(x_i)]$
 5:     Classification phase: $h_t = \mathcal{C}(B_t)$, where $h_t \in \mathcal{H}$
 6:     Let $S_t = \emptyset$
 7:     for $i = 1$ to $m$ do
 8:         Sample $x_i$ from $\begin{cases} \mathcal{U}_{D_{h_t}}, & \text{with probability } \lambda \\ \mathcal{U}_X, & \text{with probability } 1 - \lambda \end{cases}$
 9:         Let $S_t = S_t \cup \{x_i\}$
10:     end for
11:     $\tilde{x} = \mathrm{argmin}_{x\in S_t\cup\{\tilde{x}\}} f(x)$
12: end for
13: return $\tilde{x}$ and $f(\tilde{x})$
```

The classification-based optimization [25] is presented in Algorithm 1. It starts from a set of uniformly generated solutions (line 1), and then a cycle (lines 3 to 12) is followed. In each iteration, the algorithm queries the objective function to assess the generated solutions, and forms a binary classification data set B_t (line 4), where a threshold α_t is used to label the solutions as positive and negative according to $\mathrm{sign}[\alpha_t - f(x)]$. In the classification phase (line 5), a binary classifier is trained on B_t, in order to approximate the region $D_{\alpha_t} = \{x \in X \mid f(x) \leq \alpha_t\}$. During the sampling phase (lines 7 to 10), solutions are generated via sampling with probability λ from $\mathcal{U}_{D_h}$ (the uniform distribution over the positive region classified by h), i.e., setting $\mathcal{T}_h = \mathcal{U}_{D_h}$, and with the remaining probability from $\mathcal{U}_X$ (the uniform distribution over X). It is worthwhile to point out that sampling from the distribution $\mathcal{U}_{D_h}$ may imply sampling from the potential good region learned by h. The parameter λ is applied to balance the local and global search. Note that an efficient sampling from an arbitrary region is nontrivial, in our specific implementation of the classification-based optimization method that will be mentioned later, uniform sampling within the positive region D_h is straightforward and efficient. Throughout the procedure, the best-so-far solutions are recorded (line 2 and line 11), and the best one will be returned as the output solution (line 13).

C. Previous Theoretical Results

For the minimization problems, previous work [25] has derived a general (ϵ, δ)-query complexity bound of the classification-based optimization method. The bound reveals that the performance of the classification-based optimization method directly depends on two critical factors of the classification models: the error-target dependence and the shrinking rate. Since the theoretical results of this paper are based on the aforementioned general performance bound, in this section, we will introduce these two critical factors and present the general bound.

Definition 3 (Error-Target θ-Dependence) *The error-target dependence $\theta \geq 0$ of a classification-based optimization algorithm is its infimum such that, for any $\epsilon > 0$ and any t,*

$$\begin{aligned}|D_\epsilon| \cdot |D_{\alpha_t} \Delta D_{h_t}| - \theta |D_\epsilon| &\leq |D_\epsilon \cap (D_{\alpha_t} \Delta D_{h_t})| \\ &\leq |D_\epsilon| \cdot |D_{\alpha_t} \Delta D_{h_t}| + \theta |D_\epsilon|,\end{aligned}$$

where the operator Δ is the symmetric difference of two sets defined as $A_1 \Delta A_2 = (A_1 \cup A_2) - (A_1 \cap A_2)$. It characterizes, when sampling a solution x from $\mathcal{U}_X$, the dependence between the random variable that whether $x \in D_{\alpha_t} \Delta D_{h_t}$ and the random variable that whether $x \in D_\epsilon$.

Definition 3 characterizes the dependence between the classification error and the target region. The smaller θ indicates that they are more independent, and when $\theta = 0$, they are totally independent. It would be desirable that the classification error and the target region is independent, such that sampling from the positive region induced by the classifier may obtain solutions in the target region.

Definition 4 (γ-Shrinking Rate) *The shrinking rate $\gamma > 0$ of a classification-based optimization algorithm is its infimum such that $|D_{h_t}| \leq \gamma |D_{\alpha_t}|$ for all t.*

Definition 4 characterizes how large the positive region induced by the classifier. The smaller γ indicates the smaller the positive region. When the dependence between the classification error and the target region is low, it would be desirable that the positive region is small, so that the probability of sampling within the target region could be high.

Let $\mathcal{D}_t = \lambda \mathcal{U}_{D_{h_t}} + (1 - \lambda)\mathcal{U}_X$ denote the sampling distribution at iteration t, $R_{\mathcal{D}_t}$ denote the generalization error (the expected misclassification rate) of $h_t \in \mathcal{H}$ w.r.t. the target function under the distribution $\mathcal{D}_t$. Under the aforementioned two definitions, a general upper bound of the (ϵ, δ)-query complexity of the classification-based optimization method is presented below. Its proof can be found in [25].

Theorem 1 *Given a minimization problem f, $0 < \delta < 1$ and $\epsilon > 0$, if the classification-based optimization method has error-target θ-dependence and γ-shrinking rate, its (ϵ, δ)-query complexity is upper bounded by*

$$\mathcal{O}\left(\frac{1}{|D_\epsilon|}\left((1-\lambda) + \frac{\lambda}{\gamma T}\sum_{t=1}^{T}\frac{1 - Q \cdot R_{\mathcal{D}_t} - \theta}{|D_{\alpha_t}|}\right)^{-1} \ln\frac{1}{\delta}\right),$$

where $Q = 1/(1-\lambda)$.

III. Analysis on Discrete Problems

In this section, we will derive the specific (ϵ, δ)-query complexity bounds of it on some classical discrete problems. The classical problems we investigate here are the OneMax problem, the linear pseudo-Boolean functions (which contain the OneMax problem), the LeadingOnes problem, and the Trap problems.

Before presenting the theoretical analysis of this paper, we first introduce a theoretical result [26] from computational learning theory for binary classification as described in Lemma 1. Let $R_{\mathcal{D}}$ denote the generalization error (the expected misclassification rate) of $h \in \mathcal{H}$ w.r.t. the target function under distribution $\mathcal{D}$, $\widehat{R}_{\mathcal{D}}$ denote the empirical error (the misclassification rate in the seen data), $VC(\mathcal{H})$ denote the Vapnik-Chervonenkis dimension (VC-dimension) of $\mathcal{H}$. Denote the base two logarithm as $\log(\cdot)$ and the natural logarithm as $\ln(\cdot)$.

Lemma 1 *Given a fixed but unknown target function c and distribution $\mathcal{D}$, a binary labeled data set B whose m members are i.i.d. according to $\mathcal{D}$, a hypothesis space $\mathcal{H}$ with $VC(\mathcal{H}) = d$, and any $0 < \eta < 1$, then, $\forall\ h \in \mathcal{H}$, the following upper bound holds with probability at least $1 - \eta$:*

$$R_{\mathcal{D}} \leq \widehat{R}_{\mathcal{D}} + \sqrt{\frac{8}{m}\left(d \log\frac{2em}{d} + \log\frac{4}{\eta}\right)}.$$

Furthermore, if $\widehat{R}_{\mathcal{D}} = 0$, i.e., the hypothesis h is consistent with B, then, the following upper bound holds with probability at least $1-\eta$:

$$R_{\mathcal{D}} \leq \frac{2}{m}\left(d\log\frac{2em}{d} + \log\frac{2}{\eta}\right).$$

Note that we can have classification algorithms with the convergence rate of the generalization error $\widetilde{O}(\frac{1}{m})$ ignoring other variables and logarithmic terms [26], [27], where m is the sample size for the binary classification. Thus, we assume that the classification-based optimization algorithms use the classification algorithms with convergence rate $\widetilde{\Theta}(\frac{1}{m})$. For the rest of the paper, let $X = \{0,1\}^n$ with $\#X = 2^n$, and let $x_i \in \{0,1\}$ denote the i-th bit of a solution $x \in X$.

A. On OneMax Problem

We first investigate the classical OneMax problem, which is perhaps one of the deepest theoretically studied problems for EAs. In order to make the problem satisfy Definition 1, we consider the minimization version of OneMax that is described in Definition 5.

Definition 5 (OneMax Problem) *The minimization version of the OneMax problem is to find out a solution $x^* \in X$ such that $x^* = \operatorname{argmin}_{x\in X}\sum_{i=1}^{n} -x_i$.*

For the classification-based optimization method with error-target θ-dependence, γ-shrinking rate, and $\widetilde{\Theta}(\frac{1}{m})$ convergence rate, we configure it to minimize the OneMax problem as follows. First, let the sampling distribution $\mathcal{D}_t = \lambda\mathcal{U}_{D_{h_t}} + (1-\lambda)\mathcal{U}_X$. Besides, we apply the linear hypothesis function class, i.e., $h(x) = \operatorname{sign}[w^\top x + b]$, where $w \in \mathbb{R}^{n\times 1}$, $b \in \mathbb{R}$, and $w^\top$ is the transpose of w. Note that even though $x \in \{0,1\}^n$ is a binary vector, we still can apply the linear hypothesis function. Since there exists a weight vector $w = (1,1,\ldots,1)$ such that $w^\top x$ is a counter of the number of 1 bits in a solution x, the linear classification algorithm can learn the linear hypothesis function whose empirical error is zero for any fitness threshold α. It is known that, for linear hypothesis function space $\mathcal{H}$, the VC-dimension of $\mathcal{H}$ is $n+1$, i.e., $VC(\mathcal{H}) = n+1$ [26].

Theorem 2 *Given any $\epsilon > 0$ and any $0 < \delta < 1$, for the classification-based optimization method under the conditions that error-target dependence $\theta < 1$ and shrinking rate $\gamma > 0$, the (ϵ,δ)-query complexity of it for minimizing the OneMax problem belongs to*

$$\mathcal{O}\left(\left(1-\frac{\epsilon}{n}\right)n\log n\ln\frac{1}{\delta}\right).$$

Proof. According to Theorem 1, letting $Q = 2$ (i.e., $\lambda = 1/2$), it is sufficient to bound

$$\frac{1}{T}\sum_{t=1}^{T}(K_t\cdot|D_\epsilon|)/(\gamma\cdot|D_{\alpha_t}|),$$

where $K_t = 1-2R_{\mathcal{D}_t}-\theta$. Since $K_t = 1-2R_{\mathcal{D}_t}-\theta$ for all t and the classification-based optimization method is under the condition that error-target dependence $\theta < 1$, there must exist a constant $K > 0$ such that $K_t \geq K$ as long as $R_{\mathcal{D}_t} < (1-\theta)/2$ for all t. According to Lemma 1, since the empirical error $\widehat{R}_{\mathcal{D}_t} = 0$ for all t, to ensure $R_{\mathcal{D}_t} < (1-\theta)/2$ for all t, it suffices that

$$R_{\mathcal{D}_t} \leq 2m^{-1}\big(d\log(2emd^{-1}) + \log(2\eta^{-1})\big) < \frac{1-\theta}{2}.$$

Since $VC(\mathcal{H}) = n+1$ and $\ln u \leq uv + \ln\frac{1}{v} - 1$ for all $u, v > 0$, with η being a constant, we have that the required solution size in each iteration belongs to $\mathcal{O}(n)$, which will make sure that there exist a constant $K > 0$ such that $K_t \geq K$ for all t. Letting $K' = K/\gamma$, now we only need to bound

$$\frac{K'}{T}\sum_{t=1}^{T}|D_\epsilon|/|D_{\alpha_t}|.$$

For the classification-based optimization method, we set $\alpha_t = -nt/\log n$ for all t, and let the number of iterations T to approach $-nT/\log n = \epsilon - n$, where $\epsilon > 0$. Solving this equation results in that $T = (1-\epsilon/n)\log n$. For simplicity, we assume that $(1-\epsilon/n)\log n$ is a positive integer and let the classification-based method run $T = (1-\epsilon/n)\log n$ number of iterations. Now, we have that

$$\frac{K'}{T}\sum_{t=1}^{T}|D_\epsilon|/|D_{\alpha_t}| \geq \frac{K'}{(1-\epsilon/n)\log n},$$

where K' is a positive constant.

Finally, combining the inequality above with the fact that $R_{\mathcal{D}_t} < (1-\theta)/2$ for all t can be guaranteed with $\mathcal{O}(n)$ sampled solutions in each iteration and $T = (1-\epsilon/n)\log n$, by Theorem 1, the (ϵ,δ)-query complexity of the classification-based optimization method belongs to $\mathcal{O}\left((1-\epsilon/n)\,n\log n\ln\frac{1}{\delta}\right)$. ■

B. On Linear Pseudo-Boolean Functions

Actually, using the similar analysis, we can prove that Theorem 2 holds for not only the OneMax problem, but also the linear pseudo-Boolean functions.

We consider the problems that minimize general linear pseudo-Boolean functions $\sum_{i=1}^{n}\widetilde{w}_i x_i$, where $x_i \in \{0,1\}$ and $\widetilde{w}_i \in \mathbb{R}$ for all $i = 1,\ldots,n$. We assume that

$$\|\widetilde{w}\|_1 = \sum_{i=1}^{n}|\widetilde{w}_i| \leq \rho \quad \text{and} \quad \min_{i\in\{1,\ldots,n\}}|\widetilde{w}_i| > 0.$$

Therefore, $\rho > 0$ and $-\rho \leq -\|\widetilde{w}\|_1 \leq \sum_{i=1}^{n}\widetilde{w}_i x_i \leq \|\widetilde{w}\|_1 \leq \rho$. Let $\mathcal{F}_\rho = \left\{f(x) \mid f(x) = \sum_{i=1}^{n}\widetilde{w}_i x_i, \|\widetilde{w}\|_1 \leq \rho\right\}$. We define the problem of minimizing linear pseudo-Boolean functions as Definition 6.

Definition 6 (Linear Pseudo-Boolean Problem) *Given any $f \in \mathcal{F}_\rho$, the minimization version of the linear pseudo-Boolean problem is to find out a solution $x^* \in X$ such that $x^* = \operatorname{argmin}_{x\in X} f(x)$.*

For the classification-based optimization method with error-target θ-dependence, γ-shrinking rate, and $\widetilde{\Theta}(\frac{1}{m})$ convergence rate, we configure it to minimize any $f \in \mathcal{F}_\rho$ as follows. First, let the sampling distribution $\mathcal{D}_t = \lambda\mathcal{U}_{D_{h_t}} + (1-\lambda)\mathcal{U}_X$. Besides, we apply the linear hypothesis function class, i.e.,

$h(x) = \text{sign}[w^\top x + b]$. Note that even though $x \in \{0,1\}^n$ is a binary vector, we still can apply the linear hypothesis function. For any $f \in \mathcal{F}_\rho$, since there exists a weight vector $w = (w_1, w_2, \ldots, w_n)$ which is consistent with $\widetilde{w}$ in f, the linear classification algorithm can learn the linear hypothesis function whose empirical error is zero for any fitness threshold α. It is known that $VC(\mathcal{H}) = n+1$ for linear hypothesis function space $\mathcal{H}$. In each iteration of the classification-based method, we set $\alpha_t = \rho - 2\rho t/\log n$, and let the number of iterations T to approach $\rho - 2\rho t/\log n = \epsilon - \rho$. Therefore, $T = (1-\epsilon/2\rho)\log n$, where $\epsilon, \rho > 0$. Similar to the proof procedure of Theorem 2, we can derive the (ϵ,δ)-query complexity of the classification-based optimization method for minimizing the linear pseudo-Boolean functions, which is presented in Theorem 3.

Theorem 3 *Given any $f \in \mathcal{F}_\rho$, any $\epsilon > 0$ and any $0 < \delta < 1$, for the classification-based optimization method under the conditions that error-target dependence $\theta < 1$ and shrinking rate $\gamma > 0$, the (ϵ,δ)-query complexity of it for minimizing f belongs to*

$$\mathcal{O}\left(\left(1-\frac{\epsilon}{2\rho}\right)n\log n\ln\frac{1}{\delta}\right).$$

Theorem 3 indicates that the linear pseudo-Boolean functions $\mathcal{F}_\rho$ become harder for the classification-based optimization method as $\epsilon \to 0^+$ or $\rho \to +\infty$. And for any linear pseudo-Boolean functions, the (ϵ,δ)-query complexity is upper bounded by $\mathcal{O}\left(n\log n\ln\frac{1}{\delta}\right)$. Besides, it is worthwhile to point out that, applying Theorem 3, we can directly obtain that the (ϵ,δ)-query complexity of the classification-based optimization method for minimizing OneMax belongs to $\mathcal{O}((1-\frac{\epsilon}{2n})n\log n\ln\frac{1}{\delta})$. This bound is looser than that in Theorem 2, which indicates that for the specific problem we can obtain better theoretical bound.

C. On LeadingOnes Problem

In addition to the linear pseudo-Boolean functions, we also investigate the LeadingOnes problem, which is perhaps another deepest theoretically studied problem for EAs. In order to make the problem satisfy Definition 1, we consider the minimization version of LeadingOnes that is described in Definition 7.

Definition 7 (LeadingOnes Problem) *The minimization version of the LeadingOnes problem is to find out a solution $x^* \in X$ such that $x^* = \text{argmin}_{x\in X}\sum_{i=0}^{n-1}(-\prod_{j=1}^{n-i}x_j)$.*

For the classification-based optimization method with error-target θ-dependence, γ-shrinking rate, and $\widetilde{\Theta}(\frac{1}{m})$ convergence rate, we configure it to minimize the LeadingOnes problem as follows. First, let the sampling distribution $\mathcal{D}_t = \lambda\mathcal{U}_{D_{h_t}} + (1-\lambda)\mathcal{U}_X$. Besides, we apply the linear hypothesis function class, i.e., $h(x) = \text{sign}[w^\top x + b]$. Note that even though $x \in \{0,1\}^n$ is a binary vector, we still can apply the linear hypothesis function. Since there exists a weight vector

$$w = (\underbrace{1,1,\ldots,1}_{\ell},0,0,\ldots,0)$$

such that $w^\top x$ can distinguish whether the leading ℓ bits in a solution x are all 1 bits or not, the linear classification algorithm can learn the linear hypothesis function whose empirical error is zero for any fitness threshold α. For linear hypothesis function space $\mathcal{H}$, $VC(\mathcal{H}) = n+1$. Similar to the proof procedure of Theorem 2, we can derive the (ϵ,δ)-query complexity of the classification-based optimization method for minimizing the LeadingOnes problem, which is presented in Theorem 4

Theorem 4 *Given any $\epsilon > 0$ and any $0 < \delta < 1$, for the classification-based optimization method under the conditions that error-target dependence $\theta < 1$ and shrinking rate $\gamma > 0$, the (ϵ,δ)-query complexity of it for minimizing the LeadingOnes problem belongs to*

$$\mathcal{O}\left(\left(1-\frac{\epsilon}{n}\right)n\log n\ln\frac{1}{\delta}\right).$$

From Theorem 4, we observe that, the (ϵ,δ)-query complexity of the classification-based optimization method of the LeadingOnes problem is as same as that of the OneMax problem. We suppose the reason for it may be that, as same as OneMax, the high-quality region of LeadingOnes can also be learned by the linear classifier, and sampling from the learned high-quality region can lead fast convergence rate.

D. On Trap Problem

At last, we investigate the Trap problem. Actually, excluding the optimal solution from the Trap problem will make it be the OneMax problem. Thus, the (ϵ,δ)-query complexity of the classification-based optimization method can be polynomial if its configuration matches the Trap problem well. In order to make the problem satisfy Definition 1, we consider the minimization version of the Trap problem that is described in Definition 8.

Definition 8 (Trap Problem) *The minimization version of the Trap problem is to find out a solution $x^* \in X$ such that $x^* = \text{argmin}_{x\in X} -(n+1)\prod_{i=1}^{n}(1-x_i) - \sum_{i=1}^{n}x_i$.*

For the classification-based optimization method with error-target θ-dependence, γ-shrinking rate, and $\widetilde{\Theta}(\frac{1}{m})$ convergence rate, we configure it to minimize the Trap problem as follows. First, let the sampling distribution $\mathcal{D}_t = \lambda\mathcal{U}_{D_{h_t}} + (1-\lambda)\mathcal{U}_X$. Besides, we apply the linear hypothesis function class, i.e., $h(x) = \text{sign}[w^\top x + b]$. For linear hypothesis function space $\mathcal{H}$, $VC(\mathcal{H}) = n+1$. Recall that ϵ can only be chosen from the finite set $E = \{f(x) - f(x^*) \mid \forall x \in X\}$.

Theorem 5 *Given any $\epsilon > 0$ and any $0 < \delta < 1$, for the classification-based optimization method under the conditions that error-target dependence $\theta < 1$ and shrinking rate $\gamma > 0$, the (ϵ,δ)-query complexity of it for minimizing the Trap problem belongs to*

$$\mathcal{O}\left(\left(1-\frac{\epsilon}{n}\right)n\log n\ln\frac{1}{\delta}\right).$$

Proof. Since the linear hypothesis function $h(x) = \text{sign}[w^\top x + b]$ is applied in the classification-based optimization method, the classification step can not guarantee that the empirical error of h is zero. However, there exists a weight vector $w = (1, 1, \ldots, 1)$ such that $w^\top x$ is a counter of the number of 1 bits in a solution x, and thus $h(x) = \sum_{i=1}^n x_i + b$ only makes a mistake on the optimal solution $\{0, 0, \ldots, 0\}$ if $\{0, 0, \ldots, 0\}$ appears in the seen data set. To sum up, for the linear hypothesis function $h(x) = \sum_{i=1}^n x_i + b$, if its empirical error is not zero, then the optimal solution $\{0, 0, \ldots, 0\}$ must appear in the whole sampled solutions, which is an optimistic scenario and can satisfy any approximation level $\epsilon > 0$. Otherwise, the empirical error of h is zero. We do not consider this optimistic scenario, and thus the empirical error of h is zero for any fitness threshold α.

Because the optimal solution will never appear in the whole seen data set, and the empirical error of h is zero for any fitness threshold α. The (ϵ, δ)-query complexity analysis of the Trap problem is the same as that of the OneMax problem. Therefore, by Theorem 2, we can conclude the similar result that for the Trap problem, given any $\epsilon > 0$ and $0 < \delta < 1$, the classification-based optimization method can achieve the (ϵ, δ)-query complexity $\mathcal{O}\left(\left(1 - \epsilon/n\right) n \log n \ln \frac{1}{\delta}\right)$. ■

Theorem 5 indicates that, for the Trap problem, the (ϵ, δ)-query complexity of the classification-based optimization method is consistent with the expected approximation runtime of the (1+1)-EA in terms of n. The (ϵ, δ)-query complexity shows that the Trap problem is as easy as the OneMax problem, which is true if we consider only the approximation quality excluding the optimal solution.

IV. Empirical Study

We first introduce RACOS, the specific implementation of the classification-based optimization method, and we empirically study RACOS by comparing with the well-analyzed (1+1)-EA. The (1+1)-EA has been proven [28], [2] $\Theta(n \ln n)$ running time on the OneMax problem and the linear pseudo-Boolean functions, $\Theta(n^2)$ running time on the LeadingOnes problem, and exponential running time on the Trap problem. However, it should be noted that the running time would not be compared with the (ϵ, δ)-query complexity, since the latter measures the probabilistic approximation quality. The (1+1)-EA is only employed here as a reference for the empirical study.

A. Implementation of the Classification-Based Method

By equipping the randomized coordinate shrinking classification algorithm, which is depicted in Algorithm 2, into Algorithm 1 (implementing $\mathcal{C}$ in line 5), we obtain the RACOS optimization algorithm. RACOS is the specific implementation of the classification-based optimization method, and has been proposed in [25].

Algorithm 2 takes two steps: learning with randomness until all negative solutions have been excluded (lines 5-8) and shrinking (lines 9-12). Theorem 1 has revealed that the smaller error-target dependence and shrinking rate, the better the performance, which are two critical factors. Given a set of positive and negative solutions, Algorithm 2 randomly selects a dimension and collapses the dimension to the value of the positive solution until the region D_{h_t} only covers the positive solution but no negative solutions (lines 5-8). Then, lines 9-12 further shrink the positive region D_{h_t} to leave only M dimensions uncollapsed. This learning algorithm with high-level randomness achieves a positive region with a small error-target dependence and largely reduces the positive region for a small shrinking rate, which could meet two critical factors. Besides, the sampling procedure of Algorithm 1 is efficient if Algorithm 1 is equipped with Algorithm 2, since it simply draws solutions from the positive region D_{h_t} uniformly.

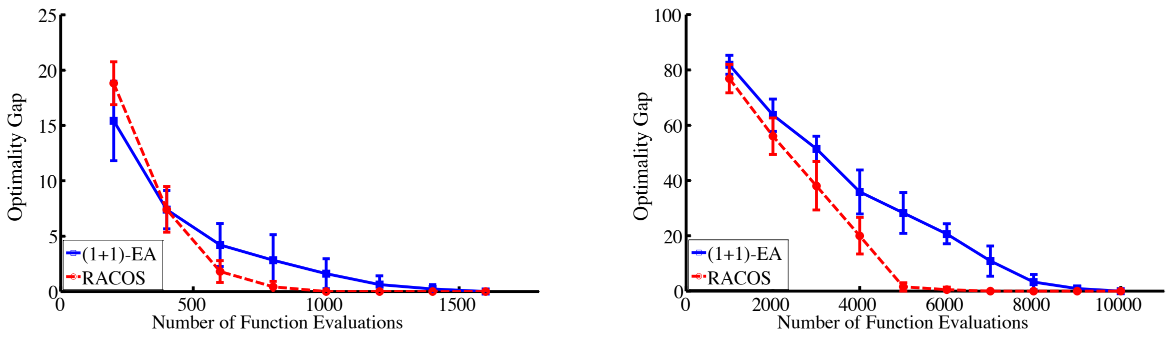
Algorithm 2 The Randomized Coordinate Shrinking Classification Algorithm for $X = \{0,1\}^n$

```
Input:
    $t$: Current iteration number;
    $B_t$: Solution set in iteration $t$;
    $I$: Index set of coordinates;
    $M \in \mathbb{N}^+$: Maximum number of uncertain coordinates.
Procedure:
 1: $B_t^+ =$ the positive solutions in $B_t$
 2: $B_t^- = B_t - B_t^+$
 3: Randomly select $x^+ = (x_1^+, \ldots, x_n^+)$ from $B_t^+$
 4: Let $D_{h_t} = X$, $I = \{1, \ldots, n\}$
 5: while $\exists x \in B_t^-$ s.t. $h_t(x) = +1$ do
 6:     $k =$ randomly selected index from the index set $I$
 7:     $D_{h_t} = D_{h_t} - \{x \in X \mid x_k \neq x_k^+\}$, $I = I - \{k\}$
 8: end while
 9: while $\#I > M$ do
10:     $k =$ randomly selected index from the index set $I$
11:     $D_{h_t} = D_{h_t} - \{x \in X \mid x_k \neq x_k^+\}$, $I = I - \{k\}$
12: end while
13: return $h_t$
```

B. Convergence Rate and Scalability

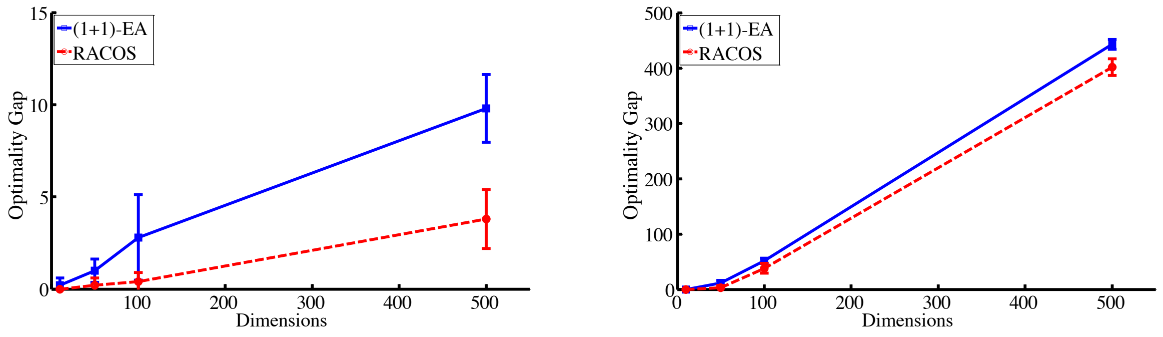
We empirically compare RACOS with the classical (1+1)-EA w.r.t. convergence rate and scalability on OneMax and LeadingOnes, respectively. For RACOS, we use the same fixed parameters in all the following experiments: in Algorithm 1, $\lambda = 0.95$, $m = 10$, and α_t is set so that only the best solution in each iteration is positive, and in Algorithm 2, $M = 2$. For (1+1)-EA, we apply the bit-wise mutation operator, i.e., flipping each bit of a solution x independently with probability $1/n$, where n is the dimension of solution space. To measure the performance of each algorithm, we apply the optimality gap, i.e., the difference of the best objective function value each algorithm finds and the optimal objective function value. The lower the optimality gap, the better the algorithm.

To study the convergence rate of the optimality gap w.r.t. the number of function evaluations, we choose $n = 100$, and set the total number of function evaluations from 2×10^2 to 1.6×10^3 for the OneMax problem and 10^3 to 10^4 for the LeadingOnes problem. Each algorithm is repeated 30 times independently, and the corresponding mean and standard deviation of the achieved optimality gap is shown in Figure 1.



(a) OneMax problem (b) LeadingOnes problem

Figure 1. Comparing the convergence rate with $n = 100$ in (a) and (b).



(a) OneMax problem (b) LeadingOnes problem

Figure 2. Comparing the scalability with $n = 10, 50, 100, 500$ in (a) and (b).

The analyzed complexity is in the order of $(1-\epsilon/n)n\log n$ indicating that the query complexity increases linearly with respect to the approximation level. In Figure 1 (a), the curve of RACOS is more convex than linear, in Figure 1 (b), its curve is close to linear. This implies that the performance of RACOS may not be as good as that in the theorems, since the theoretical study assumes a problem specific classifier rather than the general classifier in RACOS. While in both problems, (1+1)-EA converges slower than RACOS.

To study the scalability w.r.t. the solution space dimensions n, we choose n be to $10, 50, 100, 500$, and set the total number of function evaluations to be only linear to the dimensionality, i.e., $8n$ on the OneMax problem and $30n$ on the LeadingOnes problem. Each algorithm is repeated 30 times independently, and the corresponding mean and standard deviation of the achieved optimality gap is shown in Figure 2.

The analyzed complexity in the order of $(1-\epsilon/n)n\log n$ also indicates that when the number of evaluations increases only linearly, the approximation level increases sub-linearly. However, Figure 2 (a) and (b) show that the curves of RACOS increases closely linearly. While, the curves of RACOS increases slower than those of (1+1)-EA.

To sum up, RACOS performs worse than the theoretical results, where the latter are from problem specific configurations. Meanwhile, it shows better approximation performance than the well-analyzed (1+1)-EA.

V. CONCLUSIONS

This paper theoretically investigates the performance of the classification-based optimization method, a simplified version of sampling-and-classification (SAC) algorithms, in finite discrete domains, as the previous study only analyzed SAC algorithms in continuous domains.

We studied the performance of the classification-based optimization method on concrete optimization problems. Classical problems including OneMax problem, linear pseudo-Boolean functions, LeadingOnes problem, and Trap problem are employed, which have been well studied for the evolutionary algorithms. Our theoretical results disclose that the classification-based optimization method can be efficient approximation optimizer for these problems. We also empirically studied RACOS, an implementation of the classification-based method. Experimental results show that RACOS is not as good as the theoretical results, since the theoretical results are from problem specific configurations rather than the general configuration in RACOS, while it is better than the well-analyzed (1+1)-EA. In the future, we will study more conditions to disclose the power of sampling-and-learning framework, and design more efficient algorithms accordingly.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their helpful suggestions.

References

[1] J. He and X. Yao, "Drift analysis and average time complexity of evolutionary algorithms," *Artificial Intelligence*, vol. 127, no. 1, pp. 57–85, 2001.

[2] Y. Yu and Z.-H. Zhou, "A new approach to estimating the expected first hitting time of evolutionary algorithms," *Artificial Intelligence*, vol. 172, no. 15, pp. 1809–1832, 2008.

[3] B. Doerr, D. Johannsen, and C. Winzen, "Multiplicative drift analysis," *Algorithmica*, vol. 64, pp. 673–697, 2012.

[4] D. Sudholt, "A new method for lower bounds on the running time of evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 3, pp. 418–435, 2013.

[5] J. Scharnow, K. Tinnefeld, and I. Wegener, "Fitness landscapes based on sorting and shortest paths problems," in *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature (PPSN'02)*, Granada, Spain, 2002, pp. 54–63.

[6] Y. Yu, X. Yao, and Z.-H. Zhou, "On the approximation ability of evolutionary optimization with application to minimum set cover," *Artificial Intelligence*, no. 180-181, pp. 20–33, 2012.

[7] X. Yao, "Unpacking and understanding evolutionary algorithms," in *Advances in Computational Intelligence*, ser. Lecture Notes in Computer Science, J. Liu, C. Alippi, B. Bouchon-Meunier, G. Greenwood, and H. Abbass, Eds. Springer Berlin Heidelberg, 2012, vol. 7311, pp. 60–76.

[8] B. Doerr, D. Johannsen, T. Kötzing, F. Neumann, and M. Theile, "More effective crossover operators for the all-pairs shortest path problem," *Theoretical Computer Science*, vol. 471, pp. 12–26, 2013.

[9] C. Qian, Y. Yu, and Z.-H. Zhou, "An analysis on recombination in multi-objective evolutionary optimization," *Artificial Intelligence*, vol. 204, pp. 99–119, 2013.

[10] T. Storch, "On the choice of the parent population size," *Evolutionary Computation*, vol. 16, no. 4, pp. 557–578, 2008.

[11] C. Witt, "Population size versus runtime of a simple evolutionary algorithm," *Theoretical Computer Science*, vol. 403, no. 1, pp. 104–120, 2008.

[12] T. Chen, J. He, G. Sun, G. Chen, and X. Yao, "A new approach for analyzing average time complexity of population-based evolutionary algorithms on unimodal problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 5, pp. 1092–1106, 2009.

[13] T. Friedrich, J. He, N. Hebbinghaus, F. Neumann, and C. Witt, "Approximating covering problems by randomized search heuristics using multi-objective models," *Evolutionary Computation*, vol. 18, no. 4, pp. 617–633, 2010.

[14] X. Lai, Y. Zhou, J. He, and J. Zhang, "Performance analysis of evolutionary algorithms for the minimum label spanning tree problem," *IEEE Transactions on Evolutionary Computation*, vol. in press, 2014.

[15] S. Kratsch and F. Neumann, "Fixed-parameter evolutionary algorithms and the vertex cover problem," *Algorithmica*, vol. 65, no. 4, pp. 754–771, 2013.

[16] A. M. Sutton and F. Neumann, "A parameterized runtime analysis of evolutionary algorithms for the euclidean traveling salesperson problem," in *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, Toronto, Canada, 2012.

[17] T. Jansen and C. Zarges, "Fixed budget computations: A different perspective on run time analysis," in *Proceedings of the 14th International Conference on Genetic and Evolutionary Computation Conference*, Philadelphia, PA, 2012, pp. 1325–1332.

[18] F. Neumann and C. Witt, *Bioinspired Computation in Combinatorial Optimization - Algorithms and Their Computational Complexity*. Berlin, Germany: Springer-Verlag, 2010.

[19] A. Auger and B. Doerr, *Theory of Randomized Search Heuristics - Foundations and Recent Developments*. Singapore: World Scientific, 2011.

[20] T. Jansen, *Analyzing Evolutionary Algorithms*. Berlin, Germany: Springer-Verlag, 2013.

[21] P. Larrañaga and J. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Boston, MA: Kluwer, 2002.

[22] P. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[23] M. Zlochin, M. Birattari, N. Meuleau, and M. Dorigo, "Model-based search for combinatorial optimization: A critical survey," *Annals of Operations Research*, vol. 131, pp. 373–395, 2004.

[24] Y. Yu and H. Qian, "The sampling-and-learning framework: A statistical view of evolutionary algorithm," in *Proceedings of the 2014 IEEE Congress on Evolutionary Computation*, Beijing, China, 2014, pp. 149–158.

[25] Y. Yu, H. Qian, and Y.-Q. Hu, "Derivative-free optimization via classification," in *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, Phoenix, AZ, 2016.

[26] M. J. Kearns and U. V. Vazirani, *An Introduction to Computational Learning Theory*. Cambridge, MA, USA: MIT Press, 1994.

[27] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer Science & Business Media, 2000.

[28] S. Droste, T. Jansen, and I. Wegener, "On the analysis of the (1+1) evolutionary algorithm," *Theoretical Computer Science*, vol. 276, no. 1-2, pp. 51–81, 2002.