Analysis of Evolutionary Algorithms on Fitness Function With Time-Linkage Property

Weijie Zheng, Huanhuan Chen, *Senior Member, IEEE*, and Xin Yao, *Fellow, IEEE*

***Abstract*—In real-world applications, many optimization problems have the time-linkage property, that is, the objective function value relies on the current solution as well as the historical solutions. Although the rigorous theoretical analysis on evolutionary algorithms (EAs) has rapidly developed in recent two decades, it remains an open problem to theoretically understand the behaviors of EAs on time-linkage problems. This article takes the first step to rigorously analyze EAs for time-linkage functions. Based on the basic OneMax function, we propose a time-linkage function where the first bit value of the last time step is integrated but has a different preference from the current first bit. We prove that with probability $1-o(1)$, randomized local search and $(1+1)$ EA cannot find the optimum, and with probability $1-o(1)$, $(\mu+1)$ EA is able to reach the optimum.**

***Index Terms*—Convergence, evolutionary algorithms (EAs), running time analysis, time linkage.**

I. Introduction

EVOLUTIONARY algorithms (EAs), one category of stochastic optimization algorithms that are inspired by Darwinian principle and natural selection, have been widely utilized in real-world applications. Although EAs are simple and efficient to use, the theoretical understandings on the working principles and complexity of EAs are much more complicated and far behind the practical usage due to the difficulty of mathematical analysis caused by their stochastic and iterative process.

In order to fundamentally understand EAs and ultimately design efficient algorithms in practice, researchers begin the rigorous analysis on functions with simple and clear structure, majorly like pseudo-Boolean function and classic combinatorial optimization problem, like in the theory books [1]–[5]. Despite the increasing attention and insightful theoretical analyses in recent decades, there remain many important open areas that have not been considered in the evolutionary theory community.

One important open issue is about the time-linkage problem. Time-linkage problem, first introduced by Bosman [6] into the evolutionary computation community, is the optimization problem where the objective function to be optimized relies not only on the solutions of the current time but also the historical ones. In other words, the current decisions also influence the future. There are plenty of applications with time-linkage property. We just list the dynamic vehicle routing with time-varying locations to visit [6] as a slightly detailed example. Suppose that the locations are clustered. Then the current vehicle serving some locations in one cluster is more efficient to serve other locations in the same cluster instead of serving the currently available locations when the locations oscillate among different clusters in future times. Besides, the efficiency of the current vehicle routing would influence the quality of the service, which further influences the future orders, that is, future locations to visit. In a word, the current routing and the impact of it in the future together determine the income of this company. The readers could also refer to the survey in [7] to see more than 30 real-world applications, like an optimal watering scheduling to improve the quality of the crops along with the weather change [8].

The time-linkage optimization problems can be tackled offline or online according to different situations. If the problem pursues an overall solution with sufficient time budget and time-linkage dynamics can be integrated into a static objective function, then the problem can be solved offline. However, in the theoretical understanding on the static problem [1]–[5], no static benchmark function in the evolutionary theory community is time linkage.

Another situation that the real-world applications often encounter is that the solution must be solved online as the time goes by. This time-linkage online problem belongs to dynamic optimization problem [7]. As pointed out in [7], the whole evolutionary community, not only the evolutionary theory

Manuscript received April 29, 2020; revised August 12, 2020, November 16, 2020, and January 11, 2021; accepted February 16, 2021. Date of publication February 23, 2021; date of current version July 30, 2021. This work was supported in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2019A1515110177; in part by the Guangdong Provincial Key Laboratory under Grant 2020B121201001; in part by the Program for Guangdong Introducing Innovative and Enterpreneurial Teams under Grant 2017ZT07X386; in part by the Shenzhen Science and Technology Program under Grant KQTD2016112514355531; in part by the Program for University Key Laboratory of Guangdong Province under Grant 2017KSYS008; in part by the National Natural Science Foundation of China under Grant 61976111; and in part by the Science and Technology Innovation Committee Foundation of Shenzhen under Grant JCYJ20180504165652917. *(Corresponding author: Xin Yao.)*

Weijie Zheng is with the Guangdong Provincial Key Laboratory of Brain-Inspired Intelligent Computation, Research Institute of Trustworthy Autonomous Systems, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China, and also with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230052, China.

Huanhuan Chen is with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China.

Xin Yao is with the Guangdong Provincial Key Laboratory of Brain-Inspired Intelligent Computation, Research Institute of Trustworthy Autonomous Systems, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China, and also with the CERCIA, School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K. (e-mail: xiny@sustech.edu.cn).

Digital Object Identifier 10.1109/TEVC.2021.3061442

community, is lack of research on these real-world problems. The dynamic problem analyzed so far in the theory community majorly includes Dynamic OneMax [9], Magnitude and Balance [10], Maze [11], Bi-stable problem [12], dynamic linear function [13], and dynamic BinVal function [14] for dynamic pseudo-Boolean function, and dynamic combinatorial problems, including single-destination shortest path problem [15], makespan scheduling [16], vertex cover problem [17], subset selection [18], graph coloring [19], etc. However, there is no theoretical analysis on dynamic time-linkage fitness function, even no dynamic time-linkage pseudo-Boolean function is proposed for the theoretical analysis.

The main contributions of this article can be summarized as follows. This article conducts the first step toward the understanding of EAs on the time-linkage function. When solving a time-linkage problem by EAs in an offline mode, the first thing faced by the practitioners to utilize EAs is how they encode the solution. There are obviously two straightforward encoding ways. Take the objective function relying on solutions of two time steps as an example. One way is to merely ignore the time-linkage dependency by solving a nontime-linkage function with double problem size. The other way is to consider the time-linkage dependency, encode the solution with the original problem size, but store the solutions generated in the previous time steps for the fitness evaluation. When solving the time-linkage problem in an online mode, the engineers need to know before they conduct the experiments whether the algorithm they use can solve the problem or not. Hence, in this article, we design a time-linkage toy function based on OneMax to shed some light on these questions. This function, called $\text{OneMax}_{(0,1^n)}$ where n is the dimension size, is the sum of two components, one is OneMax fitness of the current n-dimensional solution, the other one is the value of the first dimension in the previous solution but multiplying minus dimension size. The design of this function considers the situation when the current solution prefers a different value from the previous solution, which could better show the influence of different encodings. Also, it could be the core element of some dynamic time-linkage functions and used in the situation that each time step we only optimize the current state of the online problem in a limited time, so that the analysis of this function could also show some insights to the undiscovered theory for the dynamic time-linkage function.

For our results, this article analyzes the theoretical behaviors of randomized local search (RLS) and two most common benchmark EAs, $(1+1)$ EA and $(\mu+1)$ EA, on $\text{OneMax}_{(0,1^n)}$. We will show that with probability $1-o(1)$, RLS and $(1+1)$ EA cannot find the optimum of $\text{OneMax}_{(0,1^n)}$ (Theorem 1) while the not small population size in $(\mu+1)$ EA can help it reach the optimum with probability $1-o(1)$ (Theorem 2). We also show that conditional on an event with probability $1-o(1)$, the expected runtime for $(\mu+1)$ EA is $O(n\mu)$ (Theorem 3).

The remainder of this article is organized as following. In Section II, we introduce the motivation and details about the designed $\text{OneMax}_{(0,1^n)}$. Section III shows the theoretical results of RLS and $(1+1)$ EA on $\text{OneMax}_{(0,1^n)}$, and our theoretical results of $(\mu+1)$ EA are shown in Section IV. Our conclusion is summarized in Section V.

II. $\text{OneMax}_{(0,1^n)}$ Function

A. $\textit{OneMax}_{(0,1^n)}$ Function

For the first time-linkage problem for theoretical analysis, we expect the function to be simple and with clear structure. OneMax, which counts the total ones in a bit string, is considered as one of the simplest pseudo-Boolean functions, and is a well-understood benchmark in the evolutionary theory community on static problems. Choosing it as a base function to add the time-linkage property could facilitate the theoretical understanding on the time-linkage property. Hence, the time-linkage function we will discuss in this article is based on OneMax. In OneMax function, each dimension has the same importance and the same preference for having a dimension value 1. We would like to show the difference, or more aggressively show the difficulty that the time-linkage property will cause, which could better help us understand the behavior of EAs on time-linkage problems. Therefore, we will introduce the solutions of the previous steps but with different importance and preference. For simplicity of analysis, we only introduce one dimension, let us say the first dimension, value of the last time step into the objective function but with the weight of $-n$, where n is the dimension size. Other weights could also be interesting, but for the first time-linkage benchmark, we just take $-n$ to show the possible difficulty caused by the time-linkage property. More precisely, this function $f:\{0,1\}\times\{0,1\}^n\to\mathbb{Z}$ is defined by

$$f(x^{t-1},x^t)=\sum_{i=1}^{n}x_i^t-nx_1^{t-1} \tag{1}$$

for two consecutive $x^{t-1}=(x_1^{t-1},\ldots,x_n^{t-1})$ and $x^t=(x_1^t,\ldots,x_n^t)\in\{0,1\}^n$. Clearly, (1) consists of two components, OneMax component relying on the current individual, and the drawing-back component determined by the first bit value of the previous individual. If our goal is to maximize (1), it is not difficult to see that the optimum is unique and the maximum value n is reached if and only if $(x_1^{t-1},x^t)=(0,1^n)$. Hence, we integrate $(0,1^n)$ and call (1) $\text{OneMax}_{(0,1^n)}$ function.

The maximization of the proposed $\text{OneMax}_{(0,1^n)}$ function specializes the maximization of the more general time-linkage pseudo-Boolean problems $h:\{0,1\}^n\times\cdots\times\{0,1\}^n\to\mathbb{R}$ defined by

$$h\left(x^{t_0},\ldots,x^{t_0+\ell}\right)=\sum_{t=0}^{\ell}h_t\left(x^{t_0+t};x^{t_0},\ldots,x^{t_0+t-1}\right) \tag{2}$$

for consecutive $x^{t_0},x^{t_0+1},\ldots,x^{t_0+\ell}$, where $\ell\in\mathbb{N}$ and could be infinite. $\text{OneMax}_{(0,1^n)}$ function could be regarded as a specialization with $\ell=1$, $h_0(x^{t_0})=-nx_1^{t_0}$, and $h_1(x^{t_0+1};x^{t_0})=\sum_{i=1}^{n}x_i^{t_0+1}$. We acknowledge that more complicated models are more interesting and practical, like with $\ell>1$, with more than one bit value and with other weight values for the historical solutions, etc., but current specialization facilitates the establishment of the first theoretical analysis for the

time-linkage problems, and our future work will focus on the analyses of more complicated models.

B. Some Notes

Time-linkage optimization problem can be solved offline or online due to different situations. In the following, we follow the main terminology from [6], the first paper that used the term "time-linkage" and introduced it into the evolutionary computation community.

1) Offline Mode: Solving the general time-linkage problem h defined above in an offline mode means that we could evaluate all possible $(x^{t_0}, x^{t_0+1}, \ldots, x^{t_0+\ell})$ before determine the final solution for the problem h. In this case, the optimum is defined differently when we use different representations. Obviously, there are two straightforward kinds of representations. One is ignoring the time-linkage fact and encoding $(x^{t_0}, x^{t_0+1}, \ldots, x^{t_0+\ell})$ into an $n(\ell+1)$-bit string as one solution, and the optimum is a search point with $n(\ell+1)$ dimensions. We denote this optimum as X^*. Since the problem is transferred to a traditional nontime-linkage problem, it is not of interest of our topic.

The other kind of representation is considering the time-linkage property and encoding $(x^{t'}, x^{t'+1}, \ldots, x^{t_0+\ell})$, $t' > t_0$, into an m-bit string, $m \in \{n, 2n, \ldots, \ell n\}$, and storing other historical solutions for objective function evaluation. In this case, the optimum is a search point with m dimensions taking the same values as the last m bit values of X^*, the optimum in the first kind of representation, condition on the stored solutions taking the same values as the corresponding bit values of X^*. For the considered $\text{OneMax}_{(0,1^n)}$ function, we encode an n-bit string as one solution and store the previous result for objective function evaluation, and the optimum is the current search point with all 1s condition on that the stored previous first bit has value 0. This representation is more interesting to us since now h and $\text{OneMax}_{(0,1^n)}$ function are truly time-linkage functions and we could figure out how EAs react to the time-linkage property. Hence, the later sections only consider this representation when solving the $\text{OneMax}_{(0,1^n)}$ function in an offline mode is analyzed.

2) Online Mode: As in [6, Sec. 2], the online mode means that we regard it as a dynamic optimization problem, and that the process continues only when the decision on the current solution is made, that is, solutions cannot be evaluated for any time $t > t^{\text{now}}$ and we can only evaluate the quality of the historical and the present solutions. More precisely, at present time t^{now}, we can evaluate

$$\tilde{h}\left(x^{t_0}, \ldots, x^{t^{\text{now}}}\right) = \sum_{t=0}^{t^{\text{now}}-t_0} \tilde{h}_t\left(x^{t_0+t}; x^{t_0}, \ldots, x^{t_0+t-1}\right)$$

where we note that $\tilde{h}_t$ could dynamically change and could be different from h_t, $t = 0, \ldots, t^{\text{now}}$ in (2) when the process ends at $t = t_0 + \ell$, since the impact of the historical solutions could change when time goes by. Usually, for the online mode, the overall optimum within the whole time period as in the offline mode cannot be reached once some nonoptimal solution is made in one time step. Hence, our goal is to obtain the function value as larger as possible before the end of the time period, or more specifically, to obtain some function value above one certain threshold. We notice the similarity to the discounted total reward in the reinforcement learning [20, Ch. 3]. However, as pointed in [6, Sec. 1], the online dynamic time-linkage problem is fundamentally different since in each time the decisions themselves are needed to be optimized and can only be made once, while in reinforcement learning the policies are the solutions and the decisions during the process serve to help determine a good policy.

Back to the $\text{OneMax}_{(0,1^n)}$, the function itself is not suitable to be solved in an online mode since it only contains the previous time step and the current step. However, we could still relate $\text{OneMax}_{(0,1^n)}$ function to online dynamic optimization problems, via regarding it as one piece of the objective function that considers the overall results during a given time period and each time step we only optimize the current piece. For example, we consider the following dynamic problem:

$$h(x,t) = \max_x \sum_{\tau=2}^{t} e^{-t+\tau-1} x_1^{\tau-2} - n x_1^{t-1} + \sum_{i=1}^{n} x_i^t \qquad (3)$$

where $x = \{x^2, \ldots, x^t\}$, $x^\tau = (x_1^\tau, \ldots, x_n^\tau) \in \{0,1\}^n$ for $\tau = 0, 1, \ldots, t$ and the initial x^0 and x^1 are given. For (3), our goal is to find the solution at some time step t when its function value is greater than $n-1$. Since the previous elements in $1, \ldots, t-2$ time steps can contribute at most $\sum_{\tau=2}^{t} e^{-t+\tau-1} \leq 1/(e-1)$ value, the goal can be transferred to find the time step when the component of the current and the last step, that is, $\text{OneMax}_{(0,1^n)}$, has the value of n. Thus, if we take the strategy for online optimization that we optimize the present each time as discussed in [6, Sec. 3], that is, for the current time t_{cur}, we optimize $h(x, t_{\text{cur}})$ with knowing $x^0, \ldots, x^{t_{\text{cur}}-1}$, then the problem can be functionally regarded as maximizing $\text{OneMax}_{(0,1^n)}$ function with n-bit string encoding as time goes by. Hence, we could reuse the optimum of $\text{OneMax}_{(0,1^n)}$ function in the offline mode as our goal for solving (3) in an online mode, and call it "optimum" for this online mode with no confusion.

In summary, considering the $\text{OneMax}_{(0,1^n)}$ problem, we note that for the representation encoding n-bit string in an offline manner and for optimizing present in an online dynamic manner, the algorithm used for these two situations are the same but with different backgrounds and descriptions of the operators. The details will be discussed when they are mentioned in Sections III and IV.

III. RLS and (1 + 1) EA Cannot Find the Optimum

A. RLS and (1 + 1) Utilized for $\text{OneMax}_{(0,1^n)}$

(1+1) EA is the simplest EA that is frequently analyzed as a benchmark algorithm in the evolutionary theory community, and RLS can be regarded as the simplification of (1 + 1) EA and thus a prestep toward the theoretical understanding of (1 + 1) EA. Both algorithms are only with one individual in their population. Their difference is on the mutation. In each generation, (1 + 1) EA employs the bit-wise mutation on the individual, that is, each bit is independently flipped with probability $1/n$, where n is the problem size, while RLS

Algorithm 1 $(1+1)$ EA/RLS to Maximize Fitness Function f Requiring Two Consecutive Time Steps

1: Generate the random initial two generations $X^0 = (X_1^0, \ldots, X_n^0)$ and $X^1 = (X_1^1, \ldots, X_n^1)$
2: **for** $g = 1, 2, \ldots$ **do**
   %% *Mutation*
3: For $(1+1)$ EA, generate $\tilde{X}^g$ via independently flipping each bit value of X^g with probability $1/n$;
   For RLS, generate $\tilde{X}^g$ via flipping one uniformly and randomly selected bit value of X^g
   %% *Selection*
4: **if** $f(X^{g-1}, X^g) > f(X^g, \tilde{X}^g)$ **then**
5: $(X^g, X^{g+1}) = (X^{g-1}, X^g)$
6: **else**
7: $(X^g, X^{g+1}) = (X^g, \tilde{X}^g)$
8: **end if**
9: **end for**

employs the one-bit mutation, that is, only one bit among n bits is uniformly and randomly chosen to be flipped. For both algorithms, the generated offspring will replace its parent as long as it has at least the same fitness as its parent.

The general RLS and $(1+1)$ EA are utilized for nontime-linkage function, and they do not consider how we choose the individual representation and do not consider the requirement to make a decision in a short time. We need some small modifications on RLS and $(1+1)$ EA to handle the time-linkage $\text{OneMax}_{(0,1^n)}$ function. The first issue, the representation choice, only happens when the problem is solved in an offline mode. As mentioned in Section II-B, for the two representation options, we only consider the one that encodes the current solution and stores the previous solution for fitness evaluation. In this encoding, we set that only offspring with better or equivalent fitness could affect the further fitness, hoping the optimization process to learn or approach to the situations which are suitable for the time-linkage property. Algorithm 1 shows our modified $(1+1)$ EA and RLS for solving $\text{OneMax}_{(0,1^n)}$, and we shall still use the name $(1+1)$ EA and RLS in this article with no confusion. In this case, the optimum of $\text{OneMax}_{(0,1^n)}$ is the 1^n as the current solution with the stored first bit value of the last generation being 0. Practically, some termination criterion is utilized in the algorithms when the practical requirement is met. Since we aim at theoretically analyzing the time to reach the optimum, we do not set the termination criterion here.

The second issue, the requirement to make a decision in a short time, happens when the problem is solved in an online mode. Detailedly, consider the problem (3) we discussed in Section II-B that in each time step we just optimize the present. If the time to make the decision is not so small that $(1+1)$ EA or RLS can solve the n-dimension problem (OneMax function), then we could obtain $X^t = (1, \ldots, 1)$ in each time step t. Obviously, in this case, the sequence of $\{X^1, \ldots, X^t\}$ we obtained will lead to a fitness less than 1 for any time step t, and thus we cannot achieve our goal and this case is not interesting. Hence, we assume the time to make the decision is small so that we cannot solve n-dimensional OneMax function, and we can just expect to find some result with better or equivalent fitness value each time step. That is, we utilize $(1+1)$ EA or RLS to solve $\text{OneMax}_{(0,1^n)}$ function, and the evolution process can go on only if some offspring with better or equivalent fitness appears. In this case, we can reuse Algorithm 1, but to note that the generation step g need not to be the same as the time step t of the fitness function since $(1+1)$ EA or RLS may need more than one generation to obtain an offspring with better or equivalent fitness for one time step.

In a word, no matter utilizing $(1+1)$ EA and RLS to solve $\text{OneMax}_{(0,1^n)}$ offline or online, in the theoretical analysis, we only consider Algorithm 1 and the optimum is the current search point with all 1s condition on that the stored previous first bit has value 0, without mentioning the solving mode and regardless of the explanations of the different backgrounds.

B. Convergence Analysis of RLS and (1 + 1) on $\text{OneMax}_{(0,1^n)}$

This section will show that with high probability RLS and $(1+1)$ EA cannot find the optimum of $\text{OneMax}_{(0,1^n)}$. Obviously, $\text{OneMax}_{(0,1^n)}$ has two goals to achieve, one is to find all 1s in the current string, and the other is to find the optimal pattern $(0,1)$ in the first bit that the current first bit value goes to 1 when the previous first bit value is 0. The two goals are somehow contradictory, so that only one individual in the population of RLS and $(1+1)$ EA will cause poor fault tolerance. Detailedly, as we will show in Theorem 1, with high probability, one of twos goal will be achieved before the optimum is found, but the population cannot be further improved.

Since this article frequently utilizes different variants of Chernoff bounds, to make it self-contained, we put them from [21], [22] into Lemma 1. Besides, before establishing our main result, with the hope that it might be beneficial for further research, we also discuss in Lemma 2 the probability estimate of the event that the increase number of ones from one parent individual to its offspring is 1 under the condition that the increase number of ones is positive in one iteration of $(1+1)$ EA, given the parent individual has a zeros.

Lemma 1 ([21], [22]): Let $\xi_1, \xi_2, \ldots, \xi_m$ be independent random variables. Let $\Xi = \sum_{i=1}^m \xi_i$:

1) if ξ_i takes values in $[0,1]$, then for all $\delta \in [0,1]$, $\Pr[\Xi \le (1-\delta)E[\Xi]] \le \exp(-\delta^2 E[\Xi]/2)$;
2) if ξ_i takes values in an interval of length c_i, then for all $\lambda \ge 0$, $\Pr[\Xi \ge E[\Xi] + \lambda] \le \exp(-2\lambda^2/\sum_{i=1}^m c_i)$;
3) if ξ_i follows the geometric distribution with success probability p for all $i \in [1..m]$, then for all $\delta \ge 0$, $\Pr[\Xi \ge (1+\delta)E[\Xi]] \le \exp(-\delta^2(m-1)/(2(1+\delta)))$; for all $\delta \in [0,1]$, $\Pr[\Xi \le (1-\delta)E[\Xi]] \le \exp(-\delta^2 m/(2-(4/3)\delta))$.

Lemma 2: Suppose $X \in \{0,1\}^n$. $Y \in \{0,1\}^n$ is generated by independently flipping each bit of X with probability $1/n$. Let $a \in [1..n]$ be the number of zeros in X, and let $|X|$ denote the number of ones in X and $|Y|$ for the number of ones in Y. Then

$$\begin{aligned}\Pr[|Y| - |X| = 1 \mid |Y| > |X|] &= \frac{\sum_{i=1}^{a} \binom{a}{i}\binom{n-a}{i-1}\frac{1}{n^{2i-1}}\left(1-\frac{1}{n}\right)^{n-2i+1}}{\sum_{i=1}^{a}\sum_{j=0}^{i-1}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}\left(1-\frac{1}{n}\right)^{n-i-j}} \\ &> 1 - \frac{ea}{n}.\end{aligned}$$

Proof: Since

$$\Pr[|Y| > |X|] = \sum_{i=1}^{a}\sum_{j=0}^{i-1}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}\left(1-\frac{1}{n}\right)^{n-i-j}$$

$$\Pr[|Y| - |X| = 1] = \sum_{i=1}^{a}\binom{a}{i}\binom{n-a}{i-1}\frac{1}{n^{2i-1}}\left(1-\frac{1}{n}\right)^{n-2i+1}$$

and $\Pr[(|Y| - |X| = 1) \cap (|Y| > |X|)] = \Pr[|Y| - |X| = 1]$, we have

$$\begin{aligned}\Pr[|Y| - |X| = 1 \mid |Y| > |X|] &= \frac{\Pr[(|Y| - |X| = 1) \cap (|Y| > |X|)]}{\Pr[|Y| > |X|]} \\ &= \frac{\sum_{i=1}^{a}\binom{a}{i}\binom{n-a}{i-1}\frac{1}{n^{2i-1}}(1-\frac{1}{n})^{n-2i+1}}{\sum_{i=1}^{a}\sum_{j=0}^{i-1}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}(1-\frac{1}{n})^{n-i-j}}.\end{aligned}$$

For $a = 1$, it is easy to see that

$$\Pr[|Y| - |X| = 1 \mid |Y| > |X|] = 1 > 1 - \frac{e}{n}.$$

For $a \geq 2$, since

$$\begin{aligned}&\sum_{i=2}^{a}\sum_{j=0}^{i-2}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}\left(1-\frac{1}{n}\right)^{n-i-j} \\ &= \sum_{i=2}^{a}\binom{a}{i}\frac{1}{n^i}\left(1-\frac{1}{n}\right)^{n-i}\sum_{j=0}^{i-2}\binom{n-a}{j}\frac{1}{n^j}\left(1-\frac{1}{n}\right)^{-j} \\ &\leq \sum_{i=2}^{a}\binom{a}{i}\frac{1}{n^i}\left(1-\frac{1}{n}\right)^{n-i}\sum_{j=0}^{i-2}\frac{(n-a)^j}{j!}\frac{1}{(n-1)^j} \\ &\leq \sum_{i=2}^{a}\binom{a}{i}\frac{1}{n^i}\left(1-\frac{1}{n}\right)^{n-i}(i-1) \\ &= \sum_{i=2}^{a}\frac{a!}{i!(a-i)!}\frac{1}{n^i}\left(1-\frac{1}{n}\right)^{n-i}(i-1) \\ &< \sum_{i=2}^{a}\frac{a!}{(i-1)!(a-i)!}\frac{1}{n^i}\left(1-\frac{1}{n}\right)^{n-i}(i-1) \\ &= \sum_{i=2}^{a}\frac{a!}{(i-2)!(a-i)!}\frac{1}{n^i}\left(1-\frac{1}{n}\right)^{n-i} \\ &= \frac{a(a-1)}{n^2}\sum_{i=2}^{a}\binom{a-2}{i-2}\frac{1}{n^{i-2}}\left(1-\frac{1}{n}\right)^{n-i} \\ &\leq \frac{a(a-1)}{n^2}\sum_{i=2}^{a}\binom{a-2}{i-2}\frac{1}{n^{i-2}}\left(1-\frac{1}{n}\right)^{a-i} \\ &= \frac{a(a-1)}{n^2}\left(\frac{1}{n}+1-\frac{1}{n}\right)^{a-2} = \frac{a(a-1)}{n^2}\end{aligned}$$

and

$$\begin{aligned}&\sum_{i=1}^{a}\sum_{j=0}^{i-1}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}\left(1-\frac{1}{n}\right)^{n-i-j} \\ &\geq \frac{a}{n}\left(1-\frac{1}{n}\right)^{n-1} > \frac{a}{en}\end{aligned}$$

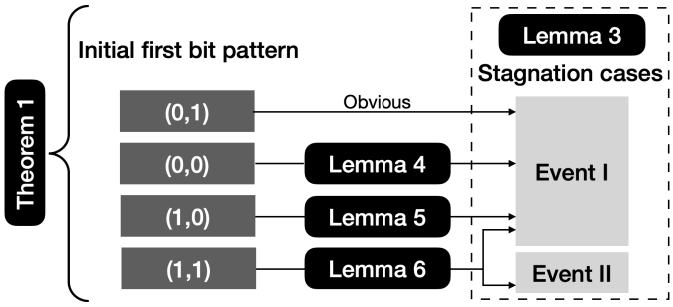

Fig. 1. Relationship between Theorem 1 and Lemmas 3–6.

we have

$$\frac{\sum_{i=2}^{a}\sum_{j=0}^{i-2}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}(1-\frac{1}{n})^{n-i-j}}{\sum_{i=1}^{a}\sum_{j=0}^{i-1}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}\left(1-\frac{1}{n}\right)^{n-i-j}} < \frac{\frac{a(a-1)}{n^2}}{\frac{a}{en}} < \frac{ea}{n}.$$

With

$$\begin{aligned}&\sum_{i=1}^{a}\binom{a}{i}\binom{n-a}{i-1}\frac{1}{n^{2i-1}}\left(1-\frac{1}{n}\right)^{n-2i+1} \\ &= \sum_{i=1}^{a}\sum_{j=0}^{i-1}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}\left(1-\frac{1}{n}\right)^{n-i-j} \\ &\quad - \sum_{i=2}^{a}\sum_{j=0}^{i-2}\binom{a}{i}\binom{n-a}{j}\frac{1}{n^{i+j}}\left(1-\frac{1}{n}\right)^{n-i-j}\end{aligned}$$

the lemma is proved. ∎

Now we show the behavior for RLS and (1+1) EA optimizing $\text{OneMax}_{(0,1^n)}$ function. The outline to establish our main theorem (Theorem 1) is shown in Fig. 1. First, Lemma 3 shows two cases once one of them happens before the optimum is reached, both RLS and $(1+1)$ EA cannot find the optimum in arbitrary further generations. Then for the nontrivial behaviors for three different initial states, Lemmas 4–6, respectively, show that the algorithm will get stuck in one of the two cases. In the following, all proofs do not specifically distinguish RLS and $(1+1)$ EA due to their similarity, and discuss each algorithm independently only when they have different behaviors. We start with one definition that will be frequently used in our proofs.

Lemma 3: Consider using $(1+1)$ EA (RLS) to optimize n-dimensional $\text{OneMax}_{(0,1^n)}$ function. Let $X^0, X^1, \ldots,$ be the solution sequence generated by the algorithm. Let:

1) *Event I:* there is a $g_0 \in \mathbb{N}$ such that $(X_1^{g_0-1}, X_1^{g_0}) = (0, 1)$ and $X_{[2..n]}^{g_0} \neq 1^{n-1}$;
2) *Event II:* there is a $g_0 \in \mathbb{N}$ such that $(X_1^{g_0-1}, X^{g_0}) = (1, 1^n)$.

Then if at a certain generation among the solution sequence, Event I or Event II happens, then $(1+1)$ EA (RLS) cannot find the optimum of $\text{OneMax}_{(0,1^n)}$ in an arbitrary long runtime afterwards.

Proof: Consider the case when Event I happens, then $(X_1^{g_0-1}, X_1^{g_0}) = (0, 1)$. In this case, the current fitness $f(X^{g_0-1}, X^{g_0}) \geq 1$. For every possible $\tilde{X}^{g_0}$, the mutation outcome of X^{g_0}, since $X_1^{g_0} = 1$, we know $f(X^{g_0}, \tilde{X}^{g_0}) \leq 0$

regardless of the values of other bits in $\tilde{X}^{g_0}$. Hence, $\tilde{X}^{g_0}$ cannot enter in the (g_0+1)th generation, that is, any progress achieved in the OneMax component of $\text{OneMax}_{(0,1^n)}$ function (any bit value changing from 0 to 1 from the 2nd to the nth bit position) cannot pass on to the next generation. Besides, $(X_1^{g_0}, X_1^{g_0+1}) = (X_1^{g_0-1}, X_1^{g_0}) = (0,1)$, then $(X_1^g, X_1^{g+1}) = (0,1)$ holds for all $g \geq g_0 - 1$. That is, RLS or $(1+1)$ EA gets stuck in this case.

Consider the case when Event II happens, that is, $(X_1^{g_0-1}, X^{g_0}) = (1, 1^n)$. In this case, the current fitness $f(X^{g_0-1}, X^{g_0}) = 0$. Similar to the above case, since $X_1^{g_0} = 1$, all possible mutation outcome $\tilde{X}^{g_0}$ along with X^{g_0} will have fitness less than or equal to 0. Hence, only when $f(X^{g_0}, \tilde{X}^{g_0}) = 0$ happens, $\tilde{X}^{g_0}$ can enter into the next generation, which means $\tilde{X}^{g_0} = 1^n$. Therefore, RLS or $(1+1)$ EA will get stuck in this case. ∎

Lemma 4: Consider using $(1+1)$ EA (RLS) to optimize n-dimensional $\text{OneMax}_{(0,1^n)}$ function. Assume that at the first generation, $(X_1^0, X_1^1) = (0,0)$ and $\sum_{j=2}^n X_j^1 < (3/4)n$. Then with probability at least $1-(e+1)n^{-1/3}$, Event I will happen at one certain generation after this initial state.

Proof: Consider the subsequent process once the number of 0-bits among $\{2, \ldots, n\}$ bit positions of the current individual, becomes less than n^c for any given constant $c < 0.5$. Note that if the first bit value changes from 0 to 1 before the number of 0-bits decreases to n^c, Event I already happens. Hence, we just consider the case that the current first bit value is still 0 at the first time when the number of the remaining 0-bits decreases to n^c. Let a denote the number of 0-bits of the current individual. We conduct the proof of this lemma based on the following two facts.

1) Among increase steps (the fitness has an absolute increase), a single increase step increases the fitness by 1 with conditional probability at least $1-ea/n$. For RLS, due to its one-bit mutation, the amount of fitness increase can only be 1 for a single increase step. For $(1+1)$ EA, Lemma 2 directly shows this fact.
2) Under the condition that one step increases the fitness by 1, with conditional probability at least $1/a$, the first bit changes its value from 0 to 1. It is obvious for RLS. For $(1+1)$ EA, suppose that the number of bits changing from 0 to 1 in this step is $m \in [1..a]$, then the probability that the first bit contributes one 0 is $\binom{a-1}{m-1}/\binom{a}{m} = m/a \geq 1/a$ for $m \geq 1$.

Note that there are $a-1$ increase steps before the remaining $n-1$ positions become all 1s if each increase step increases the fitness by 1. Then with the above two facts, it is easy to see that the probability that one individual with the $(0,1)$ first bit pattern is generated before remaining positions all have bit value 1 is at least

$$\left(\prod_{a=n^c}^{2}\left(1-\frac{ea}{n}\right)\right)\left(1-\prod_{a=n^c}^{2}\left(1-\frac{1}{a}\right)\right) = \left(\prod_{a=n^c}^{2}\left(1-\frac{ea}{n}\right)\right)\left(1-\prod_{a=n^c}^{2}\frac{a-1}{a}\right)$$

$$\geq \left(1-\frac{e}{n^{1-c}}\right)^{n^c}\left(1-\frac{1}{n^c}\right) \geq 1-\frac{e}{n^{1-2c}}-\frac{1}{n^c} \geq 1-\frac{e+1}{\min\{n^{1-2c}, n^c\}}.$$

We could just set $c = 1/3$ and obtain the probability lower bound as $1-(e+1)n^{-1/3}$. ∎

Lemma 5: Consider using $(1+1)$ EA (RLS) to optimize n-dimensional $\text{OneMax}_{(0,1^n)}$ function. Assume that at the first generation, $(X_1^0, X_1^1) = (1,0)$ and $\sum_{j=2}^n X_j^1 < (3/4)n$. Then with probability at least $1-(e+1)n^{-1/3}$, Event I will happen at one certain generation after this initial state.

Proof: Since $(X_1^0, X_1^1) = (1,0)$, we know that any offspring will have better fitness than the current individual, and will surely enter into the next generation. Then with probability $1/n$, the first bit value in the next generation becomes 1, that is, Event I happens. Otherwise, with probability $1-1/n$, it turns to the above discussed $(X_1^0, X_1^1) = (0,0)$ situation. Hence, in this situation, the probability that eventually Event I happens is at least

$$\tfrac{1}{n} + \left(1-\tfrac{1}{n}\right)\left(1-\tfrac{e+1}{n^{1/3}}\right) \geq 1-\tfrac{e+1}{n^{1/3}}.$$ ∎

Lemma 6: Consider using $(1+1)$ EA (RLS) to optimize n-dimensional $\text{OneMax}_{(0,1^n)}$ function. Assume that at the first generation, $(X_1^0, X_1^1) = (1,1)$ and $\sum_{j=2}^n X_j^1 < (3/4)n$. Then with probability at least $1-(e+1)/n^{1/3}-(n-1)\exp(-n^{1/3}/e)$, Event I or Event II will happen at one certain generation after this initial state.

Proof: For $(X_1^0, X_1^1) = (1,1)$, since in each iteration only one bit can be flipped for RLS, once the first bit is flipped from 1 to 0, the fitness of the offspring will be less than its parent and the offspring cannot enter into the next generation. Hence, for RLS, the individual will be eventually evolved to $(X_1^{g_0-1}, X^{g_0}) = (1, 1^n)$ for some $g_0 \in \mathbb{N}$. That is, Event II happens.

For $(1+1)$ EA, similar to the $(X_1^0, X_1^1) = (0,0)$ situation, we consider the subsequent process once the number of 0-bits among $\{2, \ldots, n\}$ bit positions of the current individual, becomes less than n^c for some constant $c < 0.5$, and let a denote the number of 0-bits of the current individual. If the first bit value changes from 1 to 0 before the number of 0-bits decreases to n^c, we turn to the $(X_1^0, X_1^1) = (1,0)$ situation. Otherwise, we will show that in the subsequent generations, with probability at least $1-o(1)$, the $(1,1)$ pattern will be maintained after the remaining bits reach the optimal 1^{n-1}. Consider the condition that the first bit value stays at 1s and let $\tilde{T}$ be the number of time that all $n-1$ bit positions have bit value 1 under this condition. Note that under this condition, one certain 0 bit position in $[2..n]$ does not change to 1 in t generations is at most $(1-(1-1/n)^{n-2}(1/n))^t \leq (1-1/(en))^t$. Then a union bound shows

$$\Pr\left[\tilde{T} > n^{1+c} \mid \text{the 1st value stays at 1}\right] \leq (n-1)\left(1-\frac{1}{en}\right)^{n^{1+c}} \leq (n-1)e^{-\frac{n^c}{e}}.$$

Noting that the probability that the offspring with the first bit value changing from 1 to 0 can enter into next generation is at most $(1/n)(a/n) = a/n^2 \leq 1/n^{2-c}$, we can obtain the probability that Event II happens within n^{1+c} generations is at least

$$\left(1-\frac{1}{n^{2-c}}\right)^{n^{1+c}}\left(1-(n-1)e^{-\frac{n^c}{e}}\right) \geq 1-\frac{n^{1+c}}{n^{2-c}}-(n-1)e^{-\frac{n^c}{e}} = 1-\frac{1}{n^{1-2c}}-(n-1)e^{-\frac{n^c}{e}}.$$

Further taking $c = 1/3$, together with the probability of Event I when the first bit pattern changes to $(1, 0)$ before the number of zeros decreases to n^c, we have Event I or Event II will happen with probability at least $1-(e+1)/n^{1/3}-(n-1)\exp(-n^{1/3}/e)$. ∎

Theorem 1: Let $n \geq 6$. Then with probability at least $1-(n+1)\exp(-n^{1/3}/e)-(e+1)/n^{1/3}$, $(1+1)$ EA (RLS) cannot find the optimum of the n-dimensional $\text{OneMax}_{(0,1^n)}$ function.

Proof: For the uniformly and randomly generated 1st generation, we have $E[\sum_{j=2}^{n} X_j^1] = (n-1)/2$. The Chernoff inequality in Lemma 1-2) gives that $\Pr[\sum_{j=2}^{n} X_j^1 \geq (3/4)n] \leq \exp(-(n-1)/8)$. Hence, with probability at least $1-\exp(-(n-1)/8)$, $\sum_{j=2}^{n} X_j^1 < (3/4)n$ holds, that is, $[2..n]$ bit positions have at least $(1/4)n-1$ zeros. Thus, neither Event II nor Event II happens at the first generation. We consider this initial status in the following.

Recall that from Lemma 3, Event I and Event II are two stagnation situations. For the first bit values X_1^0 and X_1^1 of the randomly generated two individuals X^0 and X^1, there are four situations, $(X_1^0, X_1^1) = (0,1), (0,0), (1,0)$, or $(1,1)$. If $(X_1^0, X_1^1) = (0,1)$, Event I already happens. Respectively from Lemmas 4–6, we could know that with probability at least $1-(e+1)/n^{1/3}-(n-1)e^{-n^{1/3}/e}$, Event I or Event II will happen in the certain generations after the initial first bit pattern $(0,0)$, $(1,0)$, and $(1,1)$.

Overall, together with the probability of $\sum_{j=2}^{n} X_j^1 < (3/4)n$, we have the probability for getting stuck is at least

$$\begin{aligned}&\left(1-e^{-\frac{n-1}{8}}\right)\left(1-\frac{e+1}{n^{1/3}}-(n-1)e^{-\frac{n^{1/3}}{e}}\right)\\ &\geq 1-e^{-\frac{n-1}{8}}-\frac{e+1}{n^{1/3}}-(n-1)e^{-\frac{n^{1/3}}{e}}\\ &\geq 1-(n+1)e^{-\frac{n^{1/3}}{e}}-\frac{e+1}{n^{1/3}}\end{aligned}$$

where the last inequality uses $\exp(-(n-1)/8) \leq 2\exp(-n/8) = 2\exp(-(n^{1/3}/e)(en^{2/3}/8)) \leq 2\exp(-n^{1/3}/e)$ when $n \geq 6$. ∎

One key reason causing the difficulty for $(1+1)$ EA and RLS is that there is only one individual in the population. As we can see in the proof, once the algorithm finds the $(0,1)$ optimal pattern in the first bit, the progress in OneMax component cannot pass on to the next generation, and once the current OneMax component finds the optimum before the first bit $(0,1)$ optimal pattern, the optimal first bit pattern cannot be obtained further. In EAs, for some cases, the large population size does not help [23], but the population could also have many benefits for ensuring the performance [24]–[27]. Similarly, we would like to know whether introducing population with not small size will improve the fault tolerance to overcome the first difficulty, and help to overcome the second difficulty since $(1, 1^n)$ individual has worse fitness so that it is easy to be eliminated in the selection. The details will be shown in Section IV.

The above analyses are conducted for the offline mode. For the online mode on problem (3), as discussed in Sections II-B and III-A, we assume the time to make the decision that could change the fitness function value is so small that we can just expect one solution with a better or equivalent fitness value at each time step. Hence, when we reuse Algorithm 1, the time t of the fitness function increases to $t+1$ once the algorithm witnesses the generation g where the generated $\tilde{X}^g$ satisfies $f(X^g, \tilde{X}^g) \geq f(X^{g-1}, X^g)$. It is not difficult to see that t and g are usually different as Algorithm 1 may need more than one generation to generate such $\tilde{X}^g$. Hence, from Theorem 1, we could also obtain that with probability at least $1-(n+1)\exp(-n^{1/3}/e)-(e+1)/n^{1/3}$, $(1+1)$ EA or RLS in our discussed online mode cannot find the solution at some time step t when the function value of problem (3) is greater than $n-1$.

IV. $(\mu+1)$ EA Can Find the Optimum

A. $(\mu+1)$ Utilized for $\text{OneMax}_{(0,1^n)}$

$(\mu+1)$ EA is a commonly used benchmark algorithm for evolutionary theory analysis, which maintains a parent population of size μ comparing with $(1+1)$ EA that has population size 1. In the mutation operator, one parent is uniformly and randomly selected from the parent population, and the bit-wise mutation is employed on this parent individual and generates its offspring. Then the selection operator will uniformly remove one individual with the worse fitness value from the union individual set of the population and the offspring.

Similar to $(1+1)$ EA discussed in Section III, the general $(\mu+1)$ EA is utilized for nontime-linkage function, and some small modifications are required for solving time-linkage problems. For solving $\text{OneMax}_{(0,1^n)}$ function in an offline mode, we just consider the representation that each individual in the population just encodes the current solution and stores the previous solution for the fitness evaluation. Similar to RLS and $(1+1)$ EA, only offspring with better or equivalent fitness could affect the further fitness, hoping the optimization process to learn the preference of the time-linkage property. Algorithm 2 shows how $(\mu+1)$ EA solves the time-linkage function that relies on two consecutive time steps. With no confusion, we shall still call this algorithm $(\mu+1)$ EA. Also note that we do not set the termination criterion in the algorithm statement, as we aim at theoretically analyzing the time to reach the optimum, that is, to generate $\tilde{X}^g = (1,\ldots,1)$ condition on that its parent X_i^g has the first bit value 0 in Algorithm 2.

We do not consider using $(\mu+1)$ EA to solve $\text{OneMax}_{(0,1^n)}$ function in an online mode. If the decision must be made in a short time period as we discussed in Section III-A, since different individuals in the parent population has their own evolving histories and different time fronts, the better offspring

Algorithm 2 $(\mu + 1)$ EA to Maximize Fitness Function f Requiring Two Consecutive Time Steps

1: Generate the random initial two generations $P^0 = \{X_1^0, \ldots, X_\mu^0\}$ and $P^1 = \{X_1^1, \ldots, X_\mu^1\}$, where $X_i^g = (X_{i,1}^g, \ldots, X_{i,n}^g), i = \{1, \ldots, \mu\}, g = \{0, 1\}$
2: **for** $g = 1, 2, \ldots$ **do**
%% Mutation
3: Uniformly and randomly select one index $\tilde{i}$ from $[1..\mu]$
4: Generate $\tilde{X}^g$ via independently flipping each bit value of $X_{\tilde{i}}^g$ with probability $1/n$
%% Selection
5: **if** $f(X_{\tilde{i}}^g, \tilde{X}^g) \geq \min_{i \in \{1,\ldots,\mu\}} \{f(X_i^{g-1}, X_i^g)\}$ **then**
6: Let $(\tilde{P}^{g-1}, \tilde{P}^g) = \{(P^{g-1}, P^g), (X_{\tilde{i}}^g, \tilde{X}^g)\}$
7: Remove the pair with the lowest fitness in $(\tilde{P}^{g-1}, \tilde{P}^g)$ uniformly at random
8: $P^{g+1} = \tilde{P}^g, P^g = \tilde{P}^{g-1}$
9: **else**
10: $P^{g+1} = P^g, P^g = P^{g-1}$
11: **end if**
12: **end for**

generated in one step cannot be regarded as the decision of the next step for the individuals in the parent population other than its own parent. If we have enough budget before time step changes, similar to the discussion in Section III-A, we will have the fitness less than 1 for any time step since $X^t = (1, \ldots, 1)$ for each time step t. Also, it is not interesting for us. Hence, the following analysis only considers $(\mu + 1)$ EA (Algorithm 2) solving $\text{OneMax}_{(0,1^n)}$ function in the offline mode.

B. Convergence Analysis of $(\mu + 1)$ on $\text{OneMax}_{(0,1^n)}$

In Section III, we show the two cases happening before the optimum is reached that cause the stagnation of $(1 + 1)$ EA and RLS for the $\text{OneMax}_{(0,1^n)}$. One is that the $(0, 1)$ first bit pattern is reached, and the other is that the current individual has the value one in all its bits with the previous first bit value as 1. The single individual in the population of $(1 + 1)$ EA or RLS results in the poor tolerance to the incorrect trial of the algorithm. This section will show that the introduction of population can increase the tolerance to the incorrect trial, and thus overcome the stagnation. That is, we will show that $(\mu + 1)$ EA can find the optimum of $\text{OneMax}_{(0,1^n)}$ with high probability. In order to give an intuitive feeling about the reason why the population can help for solving $\text{OneMax}_{(0,1^n)}$, we briefly and not-so-rigorously discuss it before establishing a rigorous analysis.

Corresponding to two stagnation cases for $(1 + 1)$ EA or RLS, $(\mu + 1)$ EA can get stuck when all individuals have the first bit value as 1 no matter the previous first bit value 0 as the first case or the previous first bit value 1 as the second case. As discussed in Section III, we know that the individual with previous first bit value as 1 has no fitness advantage against the one with previous first bit value as 0. Due to the selection operator, the one with previous first bit value 1 will be early replaced by the offspring with good fitness. As the process goes by, more detailedly in linear time of the population size in expectation, all individuals with the previous first bit value 1 will die out, and the offspring with its parent first bit value 1 cannot enter into the population. That is, the second case cannot take over the whole population to cause the stagnation.

As for the first case that the $(0, 1)$ pattern individuals takes over the population, we focus on the evolving process of the best $(0, 0)$ pattern individual, which is fertile, similar to the runtime analysis of original $(\mu + 1)$ EA in [28]. The best $(0, 0)$ pattern individuals can be incorrectly replaced only by the $(0, 1)$ pattern individual with better or the same fitness and only when all individuals with worse fitness than the best $(0, 0)$ pattern individual are replaced. With a sufficient large population size, like $\Omega(n)$ as n the problem size, with high probability, the better $(0, 1)$ pattern individuals cannot take over the whole population and the $(0, 1)$ pattern individuals with the same fitness as the best $(0, 0)$ pattern individual cannot replace all best $(0, 0)$ individuals when the population does not have any individual with worse fitness than the best $(0, 0)$ individual. That is, the first case with high probability will not happen for $(\mu+1)$ EA. In a word, the population in $(\mu+1)$ EA increases the tolerance to the incorrect trial.

Now we start our rigorous analysis. As we could infer from the above description, the difficulty of the theoretical analysis lies on the combining discussion of the intergeneration dependencies (the time-linkage among two generations) and the inner-generation dependencies (such as the selection operator). One way to handle these complicated stochastic dependencies could be the mean-field analysis, that is, mathematically analysis on a designed simplified algorithm that discards some dependencies and together with an experimental verification on the similarity between the simplified algorithm and the original one. It has been already introduced for the evolutionary computation theory [29]. However, the mean-field analysis is not totally mathematically rigorous. Hence, we do not utilize it here and analyze directly on the original algorithm. Maybe the mean-field analysis could help in more complicated algorithm and time-linkage problem, and we also hope our analysis could provide some other inspiration for the future theory work on the time-linkage problem.

For clarity, we put some calculations as lemmas in the following.

Lemma 7: Let $a, n \in \mathbb{N}$, and $a < n$. Define the functions $h_1 : [0, n - a - 1] \to (0, 1)$ and $h_2 : [1, n - a] \to (0, 1)$ by

$$h_1(d) = \frac{\binom{a+d-1}{d}}{n^{d+1}}, h_2(d) = \frac{\binom{a+d-1}{d-1}}{n^d}$$

then $h_1(d)$ and $h_2(d)$ are monotonically decreasing.

Proof: Since $h_1 > 0$, and for any $d_1, d_2 \in [0, n-a-1]$ and $d_1 \leq d_2$

$$\begin{aligned}\frac{h_1(d_1)}{h_1(d_2)} &= \frac{\binom{a+d_1-1}{d_1}}{n^{d_1+1}} \frac{n^{d_2+1}}{\binom{a+d_2-1}{d_2}} \\ &= n^{d_2-d_1} \frac{(a+d_1-1)!}{(a-1)!d_1!} \frac{(a-1)!d_2!}{(a+d_2-1)!} \\ &= n^{d_2-d_1} \frac{d_2 \cdots (d_1+1)}{(a+d_2-1) \cdots (a+d_1)} \\ &\geq n^{d_2-d_1} \left(\frac{d_1+1}{a+d_1}\right)^{d_2-d_1} \geq \left(\frac{n}{n-1}\right)^{d_2-d_1} \geq 1\end{aligned}$$

we know $h_1(d)$ is monotonically decreasing.

Similarly, since $h_1 > 0$, and for any $d_1, d_2 \in [1, n-a]$ and $d_1 \le d_2$

$$\begin{aligned}\frac{h_2(d_1)}{h_2(d_2)} &= \frac{\binom{a+d_1-1}{d_1-1}}{n^{d_1}} \frac{n^{d_2}}{\binom{a+d_2-1}{d_2-1}} \\ &= n^{d_2-d_1} \frac{(a+d_1-1)!}{a!(d_1-1)!} \frac{a!(d_2-1)!}{(a+d_2-1)!} \\ &= n^{d_2-d_1} \frac{(d_2-1)\cdots d_1}{(a+d_2-1)\cdots(a+d_1)} \\ &\ge n^{d_2-d_1}\left(\frac{d_1}{a+d_1}\right)^{d_2-d_1} \ge n^{d_2-d_1}\left(\frac{1}{n}\right)^{d_2-d_1} = 1\end{aligned}$$

we know $h_2(d)$ is monotonically decreasing. ■

Lemma 8: Let $n \in \mathbb{N}$. Define the function $g : [1, n^{1/2}] \to (0,1)$ by $g(a) = a^a/n^{a^2}$, then $g(a)$ is monotonically decreasing.

Proof: Consider $\tilde{g}(a) = \ln g(a) = a \ln a - a^2 \ln n$. For any $a_1, a_2 \in [1, n^{1/2}]$ and $a_1 < a_2$, we have

$$\begin{aligned}\tilde{g}(a_1) - \tilde{g}(a_2) &= a_1 \ln a_1 - a_1^2 \ln n - a_2 \ln a_2 + a_2^2 \ln n \\ &\ge (a_1 + a_2) \ln n - a_2 \ln a_2 > 0.\end{aligned}$$

Then, $\tilde{g}(a)$, and hence $g(a)$, are monotonically decreasing. ■

Lemma 9: Let $n > (4e)^2$. Then $(3/4)^{n^{1/2}-1} \le n^{-1/2}$.

Proof: Consider $s(n) = -(n^{1/2}-1)(1/4) + (1/2)\ln n$. Then for $n > (4e)^2 > 16$

$$s'(n) = -\tfrac{1}{4} \cdot \tfrac{1}{2} n^{-1/2} + \tfrac{1}{2} n^{-1} = \tfrac{1}{8} n^{-1}(-n^{1/2} + 4) < 0.$$

Hence, $s(n)$ is monotonically decreasing for $n > (4e)^2$. Since $s((4e)^2) = -(4e-1)/4 + \ln(4e) < 0$, we have $s(n) < 0$ for $n > (4e)^2$. Noting that $-1/4 > \ln(3/4)$, we have $(n^{1/2} - 1)\ln(3/4) < -(1/2)\ln n$ and hence $(3/4)^{n^{1/2}-1} \le n^{-1/2}$ for $n > (4e)^2$. ■

Now we show our results that $(\mu + 1)$ EA can find the optimum of $\text{OneMax}_{(0,1^n)}$ function with high probability. We start with some definitions that will be frequently used in our proofs.

Definition 10: Consider using $(\mu + 1)$ EA with population size μ to optimize n-dimensional $\text{OneMax}_{(0,1^n)}$ function. P^g is the population at the gth generation, and P^{g-1} for the $(g-1)$th generation. Let $l = \max\{f(X_i^{g-1}, X_i^g) \mid (X_{i,1}^{g-1}, X_{i,1}^g) = (0,0), X_i^{g-1} \in P^{g-1}, X_i^g \in P^g, i \in [1..\mu]\}$ be the best fitness among all individuals with the $(0,0)$ first bit pattern. For $X_i^g \in P^g, i \in [1..\mu]$:

1) it is called a *temporarily undefeated individual* if $f(X_i^{g-1}, X_i^g) > l$ and $(X_{i,1}^{g-1}, X_{i,1}^g) = (0,1)$;
2) it is called a *current front individual* if $f(X_i^{g-1}, X_i^g) = l$ and $(X_{i,1}^{g-1}, X_{i,1}^g) = (0,0)$;
3) it is called an *interior individual* if it is neither a temporarily undefeated individual nor a current front individual.

The most difficult case is when all $(1,0)$ and $(1,1)$ are replaced, which happens after linear time of the population size as in the proof of Theorem 2. Lemmas 11–13 discuss the behavior in this case. Lemma 11 will show that when the population is large enough, with high probability, there are at most $(1/2)\mu - 1$ accumulative temporarily undefeated individuals before the current front individuals have only 1 zero.

Lemma 11: Given any $\delta > 0$. For $n > (4(1+\delta)e)^2$, consider using $(\mu+1)$ EA with population size $\mu \ge 4(1+\delta)(3e+1)(n+1)$ to optimize n-dimensional $\text{OneMax}_{(0,1^n)}$ function. Assume that:

1) the current front individuals have less than $(3/4)n$ zeros;
2) all individuals are with the $(0,0)$ or $(0,1)$ pattern.

Then with probability at least $1 - \exp(-\delta^2(n-1)/(2(1+\delta)))$, there are at most $\mu/2 - 1$ accumulative temporarily undefeated individuals generated before the current front individuals only have 1 zero.

Proof: Consider the case when the current front individuals have at least 2 zeros. For the current population, let a denote the number of zeros in one current front individual, then $a \ge 2$. Let m_d denote the set of the $(0,0)$ individuals that have $a+d$ number of zeros. Obviously, m_0 is the set of the best $(0,0)$ individuals. Let A represent the event that the best $(0,0)$ fitness of the population increases in one generation, and B the event that one $(0,1)$ offspring with better fitness than the current best $(0,0)$ fitness is generated in one generation. Then we have

$$\begin{aligned}\Pr[A] &\ge \sum_{d\ge 0} \frac{|m_d|}{\mu} \frac{\binom{a+d-1}{d+1}}{n^{d+1}} \left(1 - \frac{1}{n}\right)^{n-d-1} \\ &\ge \sum_{d\ge 0} \frac{|m_d|}{e\mu} \frac{\binom{a+d-1}{d+1}}{n^{d+1}}\end{aligned}$$

and

$$\Pr[B] \le \sum_{d\ge 0} \frac{|m_d|}{\mu} \frac{\binom{a+d-1}{d}}{n^{d+1}}.$$

First, we discuss what happens under the condition that the parent is not from $m_{>a}$ individuals. Let B' represent the event that one of $m_{>a}$ individuals generates a $(0,1)$ offspring with better fitness than the current best $(0,0)$ fitness in one generation. Since

$$\frac{\binom{a+d-1}{d+1}}{\binom{a+d-1}{d}} = \frac{(a+d-1)!}{(d+1)!(a-2)!} \cdot \frac{d!(a-1)!}{(a+d-1)!} = \frac{a-1}{d+1}$$

we know

$$\begin{aligned}\frac{\Pr[A]}{\Pr[B-B']} &\ge \frac{\sum_{d=0}^{a} \frac{|m_d|}{e\mu} \frac{\binom{a+d-1}{d+1}}{n^{d+1}}}{\sum_{d=0}^{a} \frac{|m_d|}{\mu} \frac{\binom{a+d-1}{d}}{n^{d+1}}} \ge \frac{a-1}{e(a+1)} \\ &= \frac{1}{e}\left(1 - \frac{2}{a+1}\right) \ge \frac{1}{3e} \qquad (4)\end{aligned}$$

where the last inequality uses $a \ge 2$.

Second, we consider the case when the parent is selected from $m_{>a}$ individuals, that it, event B' happens. With Lemma 7, we know

$$\begin{aligned}\Pr[B'] &\le \sum_{d>a} \frac{|m_d|}{\mu} \frac{\binom{a+d-1}{d}}{n^{d+1}} \le \sum_{d>a} \frac{|m_d|}{\mu} \frac{\binom{2a-1}{a}}{n^{a+1}} \le \frac{\binom{2a-1}{a}}{n^{a+1}} \\ &= \frac{(2a-1)!}{a!(a-1)!n^{a+1}} \le \frac{(a+1)^{a-1}}{n^{a+1}} = \frac{1}{n^2}\left(\frac{a+1}{n}\right)^{a-1}.\end{aligned}$$

We discuss in two cases considering $a \geq n^c$ and $a < n^c$ for any given constant $c \in (0,1)$. From the assumption in this lemma, we know $a < (3/4)n$. Then for $a \geq n^c$, we have

$$\left(\frac{a+1}{n}\right)^{a-1} \leq \left(\frac{3}{4}\right)^{a-1} \leq \left(\frac{3}{4}\right)^{n^c-1}.$$

For $a \in [2, n^c)$

$$\left(\frac{a+1}{n}\right)^{a-1} \leq \left(\frac{n^c}{n}\right)^{a-1} \leq \frac{1}{n^{1-c}}.$$

Hence, we have for $a \in [2,(3/4)n)$, $\Pr[B'] \leq n^{c-3}$ since $(3/4)^{n^c-1} \leq n^{c-1}$ when n is large. Together with $\Pr[A] \geq 1/(e\mu n)$, we know

$$\frac{\Pr[A]}{\Pr[B']} \geq \frac{n^{2-c}}{e\mu}. \tag{5}$$

Hence, from (4) and (5), we obtain

$$\frac{\Pr[A]}{\Pr[B]} \geq \frac{1}{\frac{e\mu}{n^{2-c}} + 3e}.$$

Then if we consider the subprocess that merely consists of event A and B, we have $\Pr[A \mid A \cup B] \geq 1/(e\mu n^{c-2} + 3e + 1)$. Let X be the number of iterations that B happens in the subprocess before A occurs n times, then $E[X] \leq (e\mu n^{c-2} + 3e + 1)n$. It is not difficult to see that X is stochastically dominated by the sum of n geometric variables with success probability $1/(e\mu n^{c-2} + 3e + 1)$. Hence, with the Chernoff bound for the sum of geometric variables in Lemma 1-3), we have that for any positive constant δ

$$\Pr\left[X \geq (1+\delta)\left(\frac{e\mu}{n^{2-c}} + 3e + 1\right)n\right] \leq \exp\left(-\frac{\delta^2(n-1)}{2(1+\delta)}\right).$$

When $n > (4(1+\delta)e)^{1/(1-c)}$ and $\mu \geq 4(1+\delta)(3e+1)(n+1)$, we know

$$(1+\delta)\left(\frac{e\mu}{n^{2-c}} + 3e + 1\right)n = \frac{(1+\delta)e\mu}{n^{1-c}} + (1+\delta)(3e+1)n < \tfrac{\mu}{4} + \tfrac{\mu}{4} - 1 = \tfrac{1}{2}\mu - 1.$$

Hence, we know that with probability at least $1 - \exp(-\delta^2(n-1)/(2(1+\delta)))$, there are at most $(1/2)\mu - 1$ possible accumulative temporarily undefeated individuals before A occurs n times. Now take $c = 1/2$, then $(4(1+\delta)e)^{1/(1-c)} = (4(1+\delta)e)^2$. With $n > (4(1+\delta)e)^2 > (4e)^2$, from Lemma 9, we know $(3/4)^{n^c-1} \leq n^{c-1}$ and thus (5) hold. Noting that reducing the number of zeros in one current front individuals to 1 requires at most $n-1$ occurrence times of A, this lemma is proved. ■

Lemma 12 will show that with high probability, the current front individuals will get accumulated to $\Theta(n^{0.5})$ before all interior individuals have the same number of zeros as the current front individuals.

Lemma 12: Let $n > (4e)^2$. Consider using $(\mu+1)$ EA with population size μ to optimize n-dimensional $\text{OneMax}_{(0,1^n)}$ function.

1) Assume that:
   a) there are at most $(1/2)\mu - 1$ accumulative temporarily undefeated individuals before the current front individuals only have 1 zero;
   b) all individuals are with the $(0,0)$ or $(0,1)$ pattern.
2) Consider the phase starting from the first time when the current front individuals have $a \in [1..n]$ zeros, and ending with the first time when one $(0,0)$ pattern individual with less than a zeros is generated for $a \in [2..n]$, or ending with the first time when one $(0,1)$ pattern individual with all ones is generated for $a = 1$.

Then with probability at least $1 - \exp(-(1/20)n^{0.5})$, the current front individuals will increase by more than $(1/5)n^{0.5}$ if the current phase does not end before all individuals have at most a zeros.

Proof: Same as the notation in the proof of Lemma 11, $m_d, d \geq 0$ denotes the set of the $(0,0)$ individuals that have $a+d$ number of zeros. We now analyze the change of $|m_0|$ until all other individuals have at most a zeros, if possible, during the phase. Let C represent the event that one $(0,0)$ pattern individual with a zeros is generated in one generation, and D the event that one $(0,1)$ pattern individual with a zeros is generated in one generation. Then, we have

$$\Pr[C] \geq \sum_{d\geq 0} \frac{|m_d|}{\mu} \frac{\binom{a+d-1}{d}}{n^d} \left(1 - \frac{1}{n}\right)^{n-d} \geq \sum_{d\geq 1} \frac{|m_d|}{e\mu} \frac{\binom{a+d-1}{d}}{n^d} + \frac{|m_0|}{\mu}\left(1 - \frac{1}{n}\right)^n$$

and

$$\Pr[D] \leq \sum_{d\geq 1} \frac{|m_d|}{\mu} \frac{\binom{a+d-1}{d-1}}{n^d} + \frac{|m_0|}{\mu} \frac{\binom{a-1}{1}}{n^2}$$

where we define $\binom{0}{1} = 1$ for $a = 1$. Assume that the parent is not from $m_{>a^2}$ individuals. Let D' represent the event that one of $m_{>a^2}$ individuals generates a $(0,1)$ offspring with a zeros in one generation. Due to the definition of m_d, when $m_{>a^2}$ exist, we have $a + a^2 \leq n$, and then $a < n^{0.5}$. Since

$$\frac{\binom{a+d-1}{d}}{\binom{a+d-1}{d-1}} = \frac{(a+d-1)!}{d!(a-1)!} \cdot \frac{(d-1)!a!}{(a+d-1)!} = \frac{a}{d}$$

we know

$$\frac{\Pr[C]}{\Pr[D-D']} \geq \frac{\sum_{d=1}^{a^2} \frac{|m_d|}{e\mu} \frac{\binom{a+d-1}{d}}{n^d} + \frac{|m_0|}{\mu}\left(1-\frac{1}{n}\right)^n}{\sum_{d=1}^{a^2} \frac{|m_d|}{\mu} \frac{\binom{a+d-1}{d-1}}{n^d} + \frac{|m_0|}{\mu} \frac{\binom{a-1}{1}}{n^2}} \geq \frac{a}{ea^2} = \frac{1}{ea} > \frac{1}{en^{0.5}}. \tag{6}$$

Now we calculate $\Pr[D']$, the probability that one of $m_{>a^2}$ individual generates a $(0,1)$ offspring with a zeros in one generation, as

$$\Pr[D'] \leq \sum_{d>a^2} \frac{|m_d|}{\mu} \frac{\binom{a+d-1}{d-1}}{n^d} \leq \sum_{d>a^2} \frac{|m_d|}{\mu} \frac{\binom{a+a^2-1}{a^2-1}}{n^{a^2}} \leq \frac{\binom{a+a^2-1}{a^2-1}}{n^{a^2}} = \frac{(a+a^2-1)!}{(a^2-1)!a!n^{a^2}} \leq \frac{a^a}{n^{a^2}} \leq \frac{1}{n} \tag{7}$$

where the second inequality follows from Lemma 7 and the last inequality follows from Lemma 8 and $a \geq 1$. With

$\Pr[C] \geq (1-1/n)^n/\mu \geq 1/(2e\mu)$ for $n \geq 2$, we have

$$\frac{\Pr[C]}{\Pr[D']} \geq \frac{n}{2e\mu}. \tag{8}$$

Hence, from (6) and (8), we obtain

$$\frac{\Pr[C]}{\Pr[D]} \geq \frac{1}{\frac{2e\mu}{n} + en^{0.5}}.$$

Then if we consider the subprocess that merely consists of event C and D, we have $\Pr[C \mid C \cup D] \geq 1/((2e\mu/n) + en^{0.5} + 1)$. Recalling the definition of the phase we consider, it is not difficult to see that at the initial generation of this phase, there is only one $(0, 0)$ front individual with a number of zeros, and not difficult to see that all $(0, 1)$ individuals with a zeros, if exist, are temporarily undefeated individuals of the previous phase. Noting the assumption that there are at most $(1/2)\mu - 1$ accumulative temporarily undefeated individuals before $a = 1$, we know there are at least $(1/2)\mu$ individuals that have more than a zeros for the current phase. Hence, it requires at least $(1/2)\mu$ number of steps of the subprocess to replace these individuals with more than a zeros. Let Y be the number of times that C happens in $(1/2)\mu$ steps of the subprocess, then

$$\begin{aligned} E[Y] &\geq \frac{\frac{1}{2}\mu}{\frac{2e\mu}{n} + en^{0.5} + 1} \\ &\geq \min\left\{\frac{\frac{1}{2}\cdot 4(1+\delta)(3e+1)(n+1)}{2en^{0.5}+1}, \frac{\frac{1}{2}\mu}{2\frac{2e\mu}{n}+1}\right\} \\ &\geq \min\left\{\frac{6en}{3en^{0.5}}, \frac{1}{10e}n\right\} \geq \frac{2}{5}n^{0.5} \end{aligned}$$

where the last inequality uses $n/(10e) \geq (4en^{0.5})/(10e) = (2/5)n^{0.5}$ for $n \geq (4e)^2$. The Chernoff inequality in Lemma 1-1) gives that $\Pr[Y \leq (1/5)n^{0.5}] \leq \exp(-(1/20)n^{0.5})$. That is, with probability at least $\exp(-(1/20)n^{0.5})$, $|m_0|$ will increase by more than $(1/5)n^{0.5}$ if current phase does not end before all individuals have at most a zeros. ∎

Lemma 13 shows that with high probability, it cannot happen that all current front individuals are replaced by the $(0, 1)$ pattern individuals.

Lemma 13: Let $n > (4e)^2$. Consider using $(\mu+1)$ EA with population size μ to optimize n-dimensional $\text{OneMax}_{(0,1^n)}$ function. Considered the same assumptions and phase as in Lemma 12. Further assume that at one generation of the current phase, there are $|m_0| > (1/5)n^{0.5}$ current front individuals with a zeros, and all interior individuals has a zeros. Then with probability at least $1-(e/(e+1))^{n^{0.5}/5}$, the current phase ends before $|m_0|$ decreases to 0.

Proof: Let F denote the event that the current phase ends, that is, a $(0, 0)$ offspring with at most $a-1$ zeros when $a \geq 2$ or a $(0, 1)$ offspring with 0 zero when $a = 1$ is generated. Then

$$\begin{aligned} \Pr[F \mid a \geq 2] &\geq \tfrac{|m_0|}{e\mu}\tfrac{a-1}{n} \geq \tfrac{|m_0|}{en\mu} \\ \Pr[F \mid a = 1] &\geq \tfrac{|m_0|}{e\mu}\tfrac{1}{n} = \tfrac{|m_0|}{en\mu}. \end{aligned}$$

Then, $\Pr[F] \geq |m_0|/(en\mu)$.

Let G denote the event that a $(0, 1)$ offspring with a zeros is generated and one $(0, 0)$ individual is replaced. Suppose that the total number of the individuals with a zeros is μ'. Then

$$\Pr[G] \leq \frac{|m_0|}{\mu}\frac{1}{n}\frac{|m_0|}{\mu'+1} \leq \frac{|m_0|^2}{n\mu(\mu'+1)}.$$

Hence

$$\frac{\Pr[F]}{\Pr[G]} \geq \frac{|m_0|}{en\mu}\cdot\frac{n\mu(\mu'+1)}{|m_0|^2} = \frac{\mu'+1}{e|m_0|} \geq \frac{1}{e}$$

where the last inequality uses $\mu' \geq |m_0|$. Then

$$\Pr[G \mid F \cup G] \leq \frac{e}{e+1}.$$

Then the probability that G happens $|m_0| > (1/5)n^{0.5}$ times but F does not happen is at most $(e/(e+1))^{n^{0.5}/5}$, and this lemma is proved. ∎

Now the main result follows.

Theorem 2: Given any $\delta > 0$. Let $n > (4(1+\delta)e)^2$. Then with probability at least $1-(\mu+2)\exp(-n/8) - \exp(-\delta^2(n-1)/(2(1+\delta))) - 2n\exp(-(1/20)n^{0.5})$, $(\mu+1)$ EA with population size $\mu \geq 4(1+\delta)(3e+1)(n+1)$ can find the optimum of $\text{OneMax}_{(0,1^n)}$.

Proof: Similar to two situations in Lemma 3 that could possibly result in the stagnation of $(1 + 1)$ EA and RLS, the only two possible cases that could result in the stagnation of $(\mu + 1)$ EA are listed in the following.

1) *Event I':* There is a $g_0 \in \mathbb{N}$ such that for all $i \in [1..\mu]$, $(X_{i,1}^{g-1}, X_{i,1}^{g}) = (0, 1)$ and $X_{i,[2..n]} \neq 1^{n-1}$.
2) *Event II':* There is a $g_0 \in \mathbb{N}$ such that for all $i \in [1..\mu]$, $(X_{i,1}^{g-1}, X_{i}^{g}) = (1, 1^n)$.

For the uniformly and randomly generated P^0 and P^1, we know that the expected number of the $(1, 0)$ or $(1, 1)$ first bit patterns, that is, the expected cardinality of the set $\{i \in [1..\mu] \mid (X_{i,1}^0, X_{i,1}^1) = (1, 0) \text{ or } (1, 1)\}$, is $\mu/2$. Via the Chernoff inequality in Lemma 1-2), we know that with probability at most $1 - \exp(-\mu/8)$, at most $(3/4)\mu$ individuals have the pattern $(1, 0)$ or $(1, 1)$. Under this condition, the expected number of the $(0, 0)$ pattern is at least $(1/8)\mu$ in the whole population. The Chernoff inequality in Lemma 1-1) also gives that under the condition that at most $(3/4)\mu$ individuals have the pattern $(1, 0)$ or $(1, 1)$, with probability at least $1 - \exp(-\mu/64)$, there are at least $(1/16)\mu$ individuals with the pattern $(0, 0)$ for the initial population. Since $E[\sum_{j=1}^{n}(1 - X_{i,j}^1)] = (1/2)n$ for any $i \in [1..\mu]$, the Chernoff inequality in Lemma 1-2) on the initial population gives that $\Pr[\sum_{j=1}^{n}(1 - X_{i,j}^1) \geq (3/4)n] \leq \exp(-(1/8)n)$. Via a union bound, we know

$$\Pr\left[\exists i_0 \in [1..\mu], \sum\nolimits_{j=1}^{n}\left(1 - X_{i_0,j}^1\right) \geq \frac{3}{4}n\right] \leq \mu\exp(-\tfrac{1}{8}n).$$

Hence, noting $\mu/64 \geq (12e/64)n > (1/8)n$ from $\mu \geq 4(1 + \delta)(3e+1)(n+1)$, it is easy to see that with probability at least

$$\begin{aligned} &1 - \exp(-\tfrac{\mu}{8}) - \exp(-\tfrac{\mu}{64}) - \mu\exp(-\tfrac{1}{8}n) \\ &\geq 1 - 2\exp(-\tfrac{\mu}{64}) - \mu\exp(-\tfrac{1}{8}n) \geq 1 - (\mu+2)\exp(-\tfrac{1}{8}n) \end{aligned}$$

the initial population has at most $(3/4)\mu$ individuals with the pattern $(1, 0)$ or $(1, 1)$, at least $(1/16)\mu$ individuals with the

pattern $(0, 0)$, and $a < (3/4)n$ at the first generation. Thus, in the following, we just consider this kind of initial population.

We first show that after $O(\mu)$ generations, the individuals with the first bit pattern $(1, 0)$ or $(1, 1)$ will be replaced and will not survive in any further generation, thus Event II' cannot happen. Since $a < (3/4)n$, we know the current front individual has fitness more than 1. Note that all $(1, 0)$ or $(1, 1)$ pattern individuals have fitness at most 0. Then any offspring copied from one $(0, 0)$ pattern individual, which has the $(0, 0)$ pattern and the same fitness as its parent, will surely enter into the generation and replace some individual with the $(1, 0)$ or $(1, 1)$ pattern. Since there is at least $(1/16)\mu$ individuals with the pattern $(0, 0)$, we know that for each generation, with probability at least $1/16$, one individual with the $(1, 0)$ or $(1, 1)$ pattern will be replaced. Hence, the expected time to replace all individuals with the pattern $(1, 0)$ or $(1, 1)$ is at most $16 \cdot (3/4)\mu = 12\ \mu$ since there are at most $(3/4)\mu$ individuals with the pattern $(1, 0)$ or $(1, 1)$. Also it is not difficult to see any offspring with the $(1, 0)$ or $(1, 1)$ pattern cannot be selected into the next generation for a population only with the $(0, 0)$ or $(0, 1)$ pattern. Later, only the $(0, 1)$ and $(0, 0)$ first bit pattern can survive in the further evolution.

Now we consider Event I' after the first time when all $(1, 0)$ or $(1, 1)$ pattern individuals are replaced. From Lemma 11, we know that before $a = 1$, with probability at least $1 - \exp(-\delta^2(n-1)/(2(1+\delta)))$, there are at most $(1/2)\mu - 1$ accumulative temporarily undefeated individuals. From Lemma 12 among all possible $a \in [1..n]$, we know that if the optimum is not found before all individuals with at least 2 zeros are replaced, with probability at least $1 - n\exp(-(n^{0.5}-1)/20)$, there are at least $(1/5)n^{0.5}$ number of $(0, 0)$ individuals with $a = 1$. Then for the case $a = 1$ in Lemma 13 and considering all possible $a \in [1..n]$, we know with probability at least $1 - n(e/(e+1))^{n^{0.5}/5}$, the optimum is found.

Overall, the probability that $(\mu + 1)$ EA can find the optimum of $\text{OneMax}_{(0,1^n)}$ is at least

$$\begin{aligned}
&\left(1-(\mu+2)e^{-\frac{1}{8}n}\right)\left(1-\exp\left(-\frac{\delta^2}{2(1+\delta)}(n-1)\right)\right)\\
&\times\left(1-n\exp\left(-\tfrac{1}{20}n^{0.5}\right)\right)\left(1-n\left(\frac{e}{e+1}\right)^{\frac{1}{5}n^{0.5}}\right)\\
&\geq 1-(\mu+2)e^{-\frac{1}{8}n}-\exp\left(-\frac{\delta^2}{2(1+\delta)}(n-1)\right)\\
&-n\exp\left(-\tfrac{1}{20}n^{0.5}\right)-n\left(\frac{e}{e+1}\right)^{\frac{1}{5}n^{0.5}}\\
&\geq 1-(\mu+2)e^{-\frac{1}{8}n}-\exp\left(-\frac{\delta^2}{2(1+\delta)}(n-1)\right)\\
&-2n\exp\left(-\tfrac{1}{20}n^{0.5}\right)
\end{aligned}$$

where the last inequality uses $e^{-1/4} > e/(e+1)$. ■

In Theorem 2, we require that the population size $\mu \geq 4(1+\delta)(3e+1)(n+1)$. One may ask about the behavior when $\mu = o(n)$. We note in the proof of Lemma 11, the upper bound for the expected number of accumulative temporarily undefeated individuals is $(e\mu n^{c-2} + 3e + 1)n = \Omega(n)$. If $\mu = o(n)$, we are not able to ensure that the accumulative temporarily undefeated individuals do not take over the population in our current proof, hence, we require $\mu = \Omega(n)$.

Comparing with $(1+1)$ EA, since $(1, 1^n)$ individual, corresponding to Event II in $(1 + 1)$ EA, has no fitness advantage against the one with previous first bit value as 0, it is easy to be replaced by the offspring with previous first bit value 0 in a population. Thus, this stagnation case cannot take over the whole population to cause the stagnation of $(\mu + 1)$ EA. The possible stagnation case that the $(0, 1)$ pattern individuals take over the population, corresponding to Event I in $(1 + 1)$ EA, will not happen with a high probability because with a sufficient large, $\Omega(n)$, population size as n the problem size, with a high probability, the $(0, 0)$ pattern can be maintained until the optimum is reached, that is, the population in $(\mu + 1)$ EA increases the tolerance to the incorrect $(0, 1)$ pattern trial.

C. Runtime Analysis of $(\mu + 1)$ on $\text{OneMax}_{(0,1^n)}$

Theorem 2 only shows the probability that $(\mu + 1)$ EA can reach the optimum. One further question is about its runtime. Here, we give some comments on the runtime complexity. For the runtime of $(\mu + 1)$ EA on the original OneMax function, Witt [28] shows the upper bound of the expected runtime is $O(\mu n + n \log n)$ based on the current best individuals' replicas and fitness increasing. Analogously, for $(\mu + 1)$ EA on $\text{OneMax}_{(0,1^n)}$ function, we could consider the expected time when the number of the current front individuals with a zeros reaches n/a, that is, $|m_0| \geq n/a$, and the expected time when a $(0, 0)$ pattern offspring with less a zeros is generated for $a > 1$ or when one $(0, 1)$ pattern individual with all ones is generated for $a = 1$ conditional on that there are n/a current front individuals with a zeros. From Lemma 11, with probability at least $1-\exp(-\delta^2(n-1)/(2(1+\delta)))$, there are at most $(1/2)\mu-1$ possible accumulative temporarily undefeated individuals before the current front individuals have only 1 zero. Hence, in each generation before the current front individuals have only 1 zero, it always holds that at least half of individuals of the whole population are current front individuals and interior individuals. Hence, we could just discuss the population containing no temporarily undefeated individual and twice the upper bound of the expected time to reach the optimum as that for the true process. The $(0, 1)$ pattern offspring with a zeros will not influence the evolving process of the current front individuals we focus on until all interior individuals have a zeros. Recalling Lemma 12, we know that with probability at least $1 - \exp(-(1/20)n^{0.5})$, $|m_0| \geq (1/5)n^{0.5}$ if the current phase does not end before all individuals have at most a zeros. Hence, we just need to focus on the case when $n/a \geq (1/5)n^{0.5}$, that is, $a \leq 5n^{0.5}$.

We discuss the expected length of the phase defined in Lemma 12. When $|m_0| < n/a$, we consider the event that one replica of an m_0 individual can enter into the next generation. When the population contains interior individual(s) with more than a zeros, the probability is $|m_0|/(\mu(1-1/n)^n) \geq |m_0|/(2e\mu)$. When all interior individuals have a zeros, we require $|m_0|$ to be less than $2n/a$. Let μ'

denote the total number of the individuals with a zeros, and we know the probability of event H that one replica of an m_0 individual can enter into the next generation is $(|m_0|/\mu)(1-(1/n)^n)((\mu'+1-|m_0|)/(\mu'+1))$. Note that the probability of event G that an m_0 individual generates a $(0,1)$ pattern offspring with a zeros that successfully enters into the next generation is at most $(|m_0|/\mu)(1/n)(|m_0|/(\mu'+1))$. Then

$$\begin{aligned}\frac{\Pr[H]}{\Pr[G]} &\geq \frac{\frac{|m_0|}{\mu}\left(1-\frac{1}{n}\right)^n \frac{\mu'+1-|m_0|}{\mu'+1}}{\frac{|m_0|}{\mu}\frac{1}{n}\frac{|m_0|}{\mu'+1}} \geq \frac{1}{2e}\left(\frac{\mu'}{|m_0|}-1\right)n \\ &\geq \frac{1}{2e}\left(\frac{2(1+\delta)(3e+1)(n+1)}{|m_0|}-1\right)n \\ &\geq \frac{(1+\delta)(3e+1)a-1}{2e}n \geq \frac{3}{2}n\end{aligned}$$

where the antepenultimate inequality uses $\mu' \geq 2(1+\delta)(3e+1)(n+1)$ and the penultimate inequality uses $|m_0| < 2n/a$. Hence, $\Pr[G \mid H \cup G] \leq 2/(3n+2)$. Considering the process merely consisting of H and G, let Z be the number that H happens before G happens $(1/5)n^{0.5}$ times. Then $E[Z] \geq ((3n+2)/2-1)(1/5)n^{0.5} = (3/10)n^{1.5}$. It is not difficult to see that $Z+(1/5)n^{0.5}$ stochastically dominates the sum of $(1/5)n^{0.5}$ geometric variables with success probability $2/(3n+2)$. Hence, with the Chernoff bound for the sum of geometric variables in Lemma 1-3), we have that

$$\begin{aligned}&\Pr\left[Z+\tfrac{1}{5}n^{0.5} \leq 2n+\tfrac{2}{5}n^{0.5}\right] \\ &\leq \exp\left(-\frac{\left(1-\frac{20}{3}n^{-0.5}-\frac{4}{3}n^{-1}\right)^2\frac{1}{5}n^{0.5}}{2-\frac{4}{3}\left(1-\frac{20}{3}n^{-0.5}-\frac{4}{3}n^{-1}\right)}\right) \\ &\leq \exp\left(-\frac{\left(1-2\cdot\frac{20e+1}{3e}n^{-0.5}\right)\frac{1}{5}n^{0.5}}{2-\frac{4}{3}\left(1-\frac{20e+1}{3e}n^{-0.5}\right)}\right) \\ &\leq \exp\left(-\left(\frac{3}{1+\frac{20e+1}{3e}n^{-0.5}}-\frac{3}{2}\right)\frac{1}{5}n^{0.5}\right) \\ &\leq \exp\left(-\left(\frac{3}{1+\frac{20e+1}{3e\cdot 4e}}-\frac{3}{2}\right)\frac{1}{5}n^{0.5}\right) \leq \exp\left(-\frac{1}{20}n^{0.5}\right)\end{aligned}$$

where the second inequality uses the fact $4/(3n) \leq 4/(3\cdot 4en^{0.5}) = 1/(3en^{0.5})$ for $n > (4e)^2$ and the fact $(1-x)^2 \geq 1-2x$ for $x \in \mathbb{R}$, and the penultimate inequality uses $n > (4e)^2$. Since $2n+(1/5)n^{0.5} \geq (2n/a)+(1/5)n^{0.5}$, we know with probability at least $1-\exp(-(1/20)n^{0.5})$, $|m_0|$ could go above $2n/a$.

When $|m_0|$ goes above $2n/a$, we consider the event F that the current phase ends. Recalling the proof in Lemma 13, we know that with probability at least $1-(e/(e+1))^{n^{0.5}/5}$, F happens once before G happens $n/a \geq n^{0.5}/5$ times, thus, before $|m_0|$ goes below n/a.

In summary, in each phase, with probability at least $(1-\exp(-(1/20)n^{0.5}))(1-\exp(-(1/20)n^{0.5}))(1-(e/(e+1))^{n^{0.5}/5}) \geq 1-3\exp(-(1/20)n^{0.5})$, the current front individuals could increase its number to more than $2n/a$, and will remain above n/a afterwards. Hence, together with the runtime analysis of the original $(\mu+1)$ EA on OneMax in [28], we have the runtime result for $(\mu+1)$ EA on $\text{OneMax}_{(0,1^n)}$ in the following theorem, and know that comparing with OneMax function, the cost majorly lies on the $o(1)$ success probability for $(\mu+1)$ EA solving the time-linkage $\text{OneMax}_{(0,1^n)}$, and the asymptotic complexity remains the same for the case when $(\mu+1)$ EA is able to find the optimum.

Theorem 3: Given any $\delta > 0$. Let $n > (4(1+\delta)e)^2$. Consider using $(\mu+1)$ EA with population size $\mu \geq 4(1+\delta)(3e+1)(n+1)$ to solve the $\text{OneMax}_{(0,1^n)}$ function. Consider the same phase in Lemma 12. Let M denote the event that:

1) the first generation has at most $(3/4)\mu$ individuals with the $(1,0)$ or $(1,1)$ first bit pattern, at least $(1/4)\mu$ individuals with the $(0,0)$ pattern, and has all individuals with less than $(3/4)n$ zeros;
2) there are at most $(1/2)\mu-1$ accumulative temporarily undefeated individuals before the current front individuals only have 1 zero;
3) the number of the current front individual with a zeros can accumulate to $2n/a$ and stay above n/a if the current phase does not end.

Then event M occurring with probability at least $1-(\mu+2)\exp(-n/8)-\exp(-\delta^2(n-1)/(2(1+\delta)))-3n\exp(-(1/20)n^{0.5})$, and conditional on M, the expected runtime is $O(\mu n)$.

Recalling that the expected runtime of $(\mu+1)$ EA on OneMax is $O(\mu n + n\log n)$ [28], which is $O(n\log n)$ for $\mu = O(\log n)$. Since $\mu = \Omega(n)$ is required for the convergence on $\text{OneMax}_{(0,1^n)}$ in Section IV-B, Theorem 3 shows the expected runtime for $\text{OneMax}_{(0,1^n)}$ is $O(n^2)$ if we choose $\mu = \Theta(n)$, which is the same complexity as for OneMax with $\mu = \Theta(n)$. To this degree, comparing $(\mu+1)$ EA solving the time-linkage $\text{OneMax}_{(0,1^n)}$ with the original OneMax function, we may say the cost majorly lies on $o(1)$ convergence probability, not the asymptotic complexity.

V. Conclusion and Future Work

In recent decades, rigorous theoretical analyses on EAs has progressed significantly. However, despite that many real-world applications have the time-linkage property, that is, the objective function relies on more than one time-step solutions, the theoretical analysis on the fitness function with time-linkage property remains an open problem.

This article took the first step into this open area. We designed the time-linkage problem $\text{OneMax}_{(0,1^n)}$, which considers an opposite preference of the first bit value of the previous time step into the basic OneMax function. Via this problem, we showed that EAs with a population can prevent some stagnation in some deceptive situations caused by the time-linkage property. More specifically, we proved that the simple RLS and $(1+1)$ EA cannot reach the optimum of $\text{OneMax}_{(0,1^n)}$ with $1-o(1)$ probability but $(\mu+1)$ EA can find the optimum with $1-o(1)$ probability.

The time-linkage $\text{OneMax}_{(0,1^n)}$ problem is simple. Only the immediate previous generation and the first bit value of the historical solutions matter for the fitness function. Our future work should consider more complicated algorithms, e.g., with crossover, on more general time-linkage pseudo-Boolean functions, e.g., with more than one bit value and other weight

values for the historical solutions, and problems with practical backgrounds.

ACKNOWLEDGMENT

The authors thank Liyao Gao for his participation and discussion on the $(1+1)$ EA part during his summer intern.

REFERENCES

[1] F. Neumann and C. Witt, *Bioinspired Computation in Combinatorial Optimization–Algorithms and Their Computational Complexity*. Berlin, Germany: Springer, 2010.
[2] A. Auger and B. Doerr, *Theory of Randomized Search Heuristics: Foundations and Recent Developments*. Singapore: World Sci., 2011.
[3] T. Jansen, *Analyzing Evolutionary Algorithms: The Computer Science Perspective*. Berlin, Germany: Springer, 2013.
[4] Z.-H. Zhou, Y. Yu, and C. Qian, *Evolutionary Learning: Advances in Theories and Algorithms*. Singapore: Springer, 2019.
[5] B. Doerr and F. Neumann, *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*. Cham, Switzerland: Springer, 2020.
[6] P. A. N. Bosman, "Learning, anticipation and time-deception in evolutionary online dynamic optimization," in *Proc. Genet. Evol. Comput. Conf. GECCO Workshop*, 2005, pp. 39–47.
[7] T. T. Nguyen, "Continuous dynamic optimisation using evolutionary algorithms," Ph.D. dissertation, Sch. Comput. Sci., Univ. Birmingham, Birmingham, U.K., 2011.
[8] T. Morimoto, Y. Ouchi, M. Shimizu, and M. Baloch, "Dynamic optimization of watering satsuma mandarin using neural networks and genetic algorithms," *Agr. Water Manag.*, vol. 93, nos. 1–2, pp. 1–10, 2007.
[9] S. Droste, "Analysis of the $(1+1)$ EA for a dynamically changing ONEMAX-variant," in *Proc. Congr. Evol. Comput. (CEC)*, vol. 1, 2002, pp. 55–60.
[10] P. Rohlfshagen, P. K. Lehre, and X. Yao, "Dynamic evolutionary optimisation: An analysis of frequency and magnitude of change," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, 2009, pp. 1713–1720.
[11] T. Kötzing and H. Molter, "ACO beats EA on a dynamic pseudo-Boolean function," in *Proc. Int. Conf. Parallel Problem Solving Nat. (PPSN)*, 2012, pp. 113–122.
[12] T. Jansen and C. Zarges, "Analysis of randomised search heuristics for dynamic optimisation," *Evol. Comput.*, vol. 23, no. 4, pp. 513–541, 2015.
[13] J. Lengler and U. Schaller, "The $(1+1)$-EA on noisy linear functions with random positive weights," in *Proc. IEEE Symp. Series Comput. Intell. (SSCI)*, 2018, pp. 712–719.
[14] J. Lengler and J. Meier, "Large population sizes and crossover help in dynamic environments," 2020. [Online]. Available: arXiv:2004.09949.
[15] A. Lissovoi and C. Witt, "Runtime analysis of ant colony optimization on dynamic shortest path problems," *Theor. Comput. Sci.*, vol. 561, pp. 73–85, Jan. 2015.
[16] F. Neumann and C. Witt, "On the runtime of randomized local search and simple evolutionary algorithms for dynamic makespan scheduling," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2015, pp. 3742–3748.
[17] M. Pourhassan, W. Gao, and F. Neumann, "Maintaining 2-approximations for the dynamic vertex cover problem using evolutionary algorithms," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, 2015, pp. 903–910.
[18] V. Roostapour, A. Neumann, F. Neumann, and T. Friedrich, "Pareto optimization for subset selection with dynamic cost constraints," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2019, pp. 2354–2361.
[19] J. Bossek, F. Neumann, P. Peng, and D. Sudholt, "Runtime analysis of randomized search heuristics for dynamic graph coloring," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, 2019, pp. 1443–1451.
[20] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
[21] B. Doerr, "Analyzing randomized search heuristics: Tools from probability theory," in *Theory of Randomized Search Heuristics: Foundations and Recent Developments*. Singapore: World Sci., 2011, pp. 1–20.
[22] B. Doerr, "Probabilistic tools for the analysis of randomized optimization heuristics," in *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, B. Doerr and F. Neumann, Eds. Cham, Switzerland: Springer, 2020, pp. 1–87.
[23] T. Chen, K. Tang, G. Chen, and X. Yao, "A large population size can be unhelpful in evolutionary algorithms," *Theor. Comput. Sci.*, vol. 436, pp. 54–70, Jun. 2012.
[24] J. He and X. Yao, "From an individual to a population: An analysis of the first hitting time of population-based evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 6, no. 5, pp. 495–511, Oct. 2002.
[25] D.-C. Dang, T. Jansen, and P. K. Lehre, "Populations can be essential in tracking dynamic optima," *Algorithmica*, vol. 78, no. 2, pp. 660–680, 2017.
[26] D. Corus and P. S. Oliveto, "On the benefits of populations for the exploitation speed of standard steady-state genetic algorithms," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, 2019, pp. 1452–1460.
[27] D. Sudholt, "The benefits of population diversity in evolutionary algorithms: A survey of rigorous runtime analyses," in *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, B. Doerr and F. Neumann, Eds. Cham, Switzerland: Springer, 2020, pp. 359–404.
[28] C. Witt, "Runtime analysis of the $(\mu+1)$ EA on simple pseudo-Boolean functions," *Evol. Comput.*, vol. 14, no. 1, pp. 65–86, 2006.
[29] B. Doerr and W. Zheng, "Working principles of binary differential evolution," *Theor. Comput. Sci.*, vol. 801, pp. 110–142, Jan. 2020.

Weijie Zheng received the bachelor's degree in mathematics and applied mathematics from the Harbin Institute of Technology, Harbin, China, in July 2013, and the Doctoral degree in computer science and technology from Tsinghua University, Beijing, China, in October 2018.

He is currently a Postdoctoral Researcher with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China, and the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. His current research majorly focuses on the theoretical analysis and design of evolutionary algorithms, especially binary differential evolution, and estimation-of-distribution algorithms.

Huanhuan Chen (Senior Member, IEEE) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 2004, and the Ph.D. degree in computer science from the University of Birmingham, Birmingham, U.K., in 2008.

He is currently a Full Professor with the School of Computer Science and Technology, USTC. His current research interests include neural networks, Bayesian inference, and evolutionary computation.

Prof. Chen was a recipient of the 2015 International Neural Network Society Young Investigator Award, the 2012 IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation Award, the IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and the 2009 British Computer Society Distinguished Dissertations Award. He is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS and the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE.

Xin Yao (Fellow, IEEE) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 1982, the M.Sc. degree from the North China Institute of Computing Technologies, Langfang, China, in 1985, and the Ph.D. degree from USTC in 1990.

He is a Chair Professor of Computer Science with the Southern University of Science and Technology, Shenzhen, China, and a part-time Professor of Computer Science with the University of Birmingham, Birmingham, U.K. His major research interests include evolutionary computation, ensemble learning, and their applications to software engineering.

Dr. Yao received the prestigious Royal Society Wolfson Research Merit Award in 2012, the IEEE CIS Evolutionary Computation Pioneer Award in 2013, and the 2020 IEEE Frank Rosenblatt Award. His work won the 2001 IEEE Donald G. Fink Prize Paper Award, the 2010, 2016, and 2017 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Awards, the 2011 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and many other best paper awards at conferences. He was a Distinguished Lecturer of the IEEE Computational Intelligence Society. He was the President of IEEE CIS from 2014 to 2015 and the Editor-in-Chief of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2003 to 2008.