IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 29 May 2024, accepted 11 June 2024, date of publication 24 June 2024, date of current version 22 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3417952

RESEARCH ARTICLE

Adaptive Racing Sampling Based Immune Optimization Approach for Nonlinear Multi-Objective Chance Constrained Programming

KAI YANG[1,2] **AND RENCHONG ZHANG**[3]

[1]Department of Information Engineering, Guizhou Communications Polytechnic, Guiyang 551400, China
[2]Guizhou Big Data Academy, Guizhou University, Guiyang 550025, China
[3]Computer and Information Engineering College, Guizhou University of Commerce, Guiyang 550014, China

Corresponding author: Kai Yang (kyang2022@tom.com)

This work was supported in part by the Science and Technology Foundation of Guizhou (Guizhou-Science-Contract) under Grant [2019]1178 and Grant [2020]1Y423, in part by Guizhou Sci-Tech Cooperation Platform Talent under Grant CXTD[2021]008, and in part by the Science and Technology Project of the Department of Transportation of Guizhou Province under Grant 2022-122-020 and Grant 2022-321-013.

ABSTRACT This work investigates a multi-objective immune optimization approach to solve the general type of nonlinear multi-objective chance constrained programming without prior noise information. One such kind of model is first converted into a sample-dependent approximation one, while a sample bound estimate model is theoretically acquired based on the empirical Bernstein bound, in order to control the sampling size of random variable. Secondly, a feasibility detection approach with adaptive sampling is designed to quickly justify whether an individual is empirically feasible. Inspired by the danger theory, an artificial immune optimization model is drawn in terms of immune response mechanisms in the immune system, which derives out a multi-objective chance constrained optimizer with small populations and multiple evolutionary strategies. The computational complexity of the optimizer depends mainly on the sample bound and the size of memory pool. Comparative experiments have validated that it is a robust, stable, and effective optimizer with high efficiency while helping for solving complex chance constrained problems.

INDEX TERMS Multi-objective chance constrained programming, immune optimization, adaptive sampling, danger theory, sample-dependent approximation.

I. INTRODUCTION

Chance constrained programming (CCP) is a typical topic of stochastic programming, proposed early by Charnes and Cooper in 1959 [1]. Since its feasible region is usually nonconvex, it is almost impossible to find the theoretical optimal solution by means of mathematical programming techniques. Therein, many researchers made great efforts to study intelligent optimization methodologies so as to gain the approximate optimal solution [2], [3], [4], [5], [44], in which the concerns on sample bound estimate and constraint handling become extremely important. Generally, when random variables are with normal distributions, CCP can be transformed equivalently into an analytically deterministic model, and meanwhile some existing numerical iterative approaches and intelligent optimization techniques can effectively find its approximate solution(s) [6], [7], [8]. Conversely, once such uncertain factors are with unknown distribution information or known complex distribution features, several kinds of approaches, e.g., convex approximation [9], robust optimization [10], scenario approximation [11], sample average approximation [12], [13], can change CCP models into

The associate editor coordinating the review of this manuscript and approving it for publication was Daniel Augusto Ribeiro Chaves.

approximation ones, and subsequently intelligent approaches as a competitive tool can be borrowed to find their approximate solutions.

Multi-objective chance-constrained programming (MCCP) as a provoking topic of CCP is to simultaneously optimize multiple conflicting sub-objectives under the limits of chance constraints. Conventional optimization techniques face great challenges, due to the difficulties of individual importance discrimination and constraint handling. This activates us to successively probe into advanced multi-objective stochastic optimization techniques for MCCP problems from the viewpoint of intelligent optimization. The main concern includes three points: (i) high-effective individual dominance in stochastic multi-objective environments, (ii) high-efficient individual sampling, and (iii) advanced evolutionary models. The main challenge consists in that, besides MCCP's feasible region is usually nonconvex, the stochastic factors perturb seriously a search procedure to identify those non-dominated solutions. This results that only few achievements have been reported in the literature [14], [15], and consequently many studies on MCCP still remain open. Especially, new multi-objective chance-constrained optimization techniques are desired, due to wide engineering application background. Based on such consideration, our main contribution in the present work includes three points: (i) a sample bound estimate model is theoretically acquired to control the maximal sampling size of random variable, (ii) a feasibility detection approach is designed to handle MCCP's chance constraints, (iii) a new artificial immune optimization model and subsequently a related adaptive racing sampling-based multi-objective immune optimization approach (ARSMIOA) are developed, inspired by the danger theory in immunology.

II. RELATED WORK SURVEY ON CCP

A. CHANCE CONSTRAINT-HANDING

Usually, chance constraints are represented by probabilistic equalities or inequalities. When their random variables are with known distribution information, they can be equivalently changed into analytic constraints. For example, Deb et al. [16] converted this kind of random variable into a normally distributed random variable, and hence the chance constraints were handled by single- or decouple-loop optimization techniques. Unfortunately, such constraint transformation will cause high computational cost; especially, when a CCP problem includes multiple chance constraints, it will be equivalently transformed into a nested programming problem with multiple sub- optimization models and accordingly, the high computational complexity occurs inevitably. Thus, such model transformation is not available generally, and hence chance constraint approximation handling becomes popular. Model approximation is a usual way to transform CCP into a sample average approximation (SAA, [12]) model, and some state-of-the-art static optimization approaches can be adopted to solve it. However, SAA requires that any candidate solutions be with the same large sample size [8], which necessarily causes high computational complexity. Thereby, it becomes extremely important to discuss sample-allocation or adaptive sampling approaches [17], [18] for which the sample size of a random variable depends on the quality of each candidate solution, namely excellent candidate solutions can attach large sample sizes and conversely those poor ones can only be assigned few samples. Additionally, in order to clearly know the relation of approximation between CCP and the related SAA, studies on sample bound estimate have been well done. Luedtke and Shapiro et al. [12], [19] investigated the relation between solutions for single-objective chance constrained programming and the related SAA in terms of the probability theory or probabilistic inequalities, by which some valuable sample bound estimate models were discovered. Their valuable sample bounds can be used to control the maximal sampling sizes of random variables.

B. INTELLIGENT OPTIMIZATION ON CCP

As we mentioned above, when random variables are with known distribution information, CCP can be converted into a static model and later solved by classical optimization approaches. Conversely, usually it can only be replaced by approximation models, and handled by intelligent optimization techniques [8], [20], [21], [22], [23], [43]. For instance, Liu et al [8] investigated its approximation model by replacing its chance constraints and objective function with RBF networks, and subsequently the conventional steady genetic algorithm was adopted to seek its approximate solution. Poojari et al. [20] developed two similar steady genetic algorithms with a static sampling rule for a general type of single-objective CCP, after designing two scoring functions used for evaluating the quality of individual. Luedtke [23] suggested an improved branch-and-bound method for solving a type of single-objective CCP problem with discrete distribution, finite support and stochastic multi-sided constraints. One such approach is effective in certain limits, but its application scope is narrow. Additionally, CCP has also been well investigated in real-world engineering application, since a large number of engineering problems can be modeled into CCP models [24], [25], [26], [27], [28], [29], [30], [31], [32], [33]. This also motivates researchers to study effective and efficient intelligent techniques for seeking their decision schemes. For example, Tian et al. [32] designed a hybrid optimization approach to process a disassembly sequence planning problem with stochastic factors, after integrating stochastic simulation and RBF networks with genetic algorithms.

Since multiple conflicting performance indices and chance constraints, MCCP becomes extremely difficult in finding its non-dominated solutions. To this point, only a few pioneering works have been reported in the literature [29], [30], [31], [32], [33], [34]. The main concern concentrates on addressing the problems of individual's dominance, diversity maintenance, adaptive sampling and population evolution. Liu et al. [30] developed a multi-objective algorithm to solve a bi-objective chance-constrained programming problem, based on sequential optimization, Latin hypercube sampling and the idea of parallel processing. Nikokalam-Mozafar et al. [31] converted a single-allocation hub covering problem with the variable capacity and uncertainty parameters into a bi-objective chance constrained programming model with normally distributed random variables. One such problem is required to simultaneously minimize the total transportation cost and the transportation time from an origin to a destination, solved by a discrete multi-objective invasive weed optimization approach. Kamjoo et al. [34] designed a MCCP model which was used to formulate a hybrid renewable energy system, and coped with it by stochastic simulation-based NSGA-II.

III. PROBLEM FORMULATION AND TRANSFORMATION

Consider the following nonlinear MCCP problem of the form

$$\text{MCCP}\ \min_{x\in D}\ E\left[f_1\left(\boldsymbol{x},\xi\right),\cdots,f_m\left(\boldsymbol{x},\xi\right)\right]$$
$$s.t.,\begin{cases} p_i(\boldsymbol{x})\equiv\Pr\left\{G_i\left(\boldsymbol{x},\eta\right)\le 0\right\}\ge\alpha_i,\\ g_j\left(\boldsymbol{x}\right)\le 0,h_k\left(\boldsymbol{x}\right)=0,\\ 1\le i\le I,1\le j\le J,1\le k\le K,\end{cases}$$

with bounded and closed decision domain D in $R^p, x \in D$, confidence levels α_i, and unknown distributional random vectors ξ and η in R^q, where $f_i(.,\xi)$ is the i-th stochastic sub-objective function;$G_i(.,\xi)$ is the i-th stochastic constraint function; $g_j(.)$ and $h_k(.)$ denote the deterministic constraint functions;$E[.]$ is the expected operator, namely

$$E[f_1(\boldsymbol{x},\xi),\ldots,f_m(\boldsymbol{x},\xi)]=(E[f_1(\boldsymbol{x},\xi)],\ldots,E[f_m(\boldsymbol{x},\xi)];$$

Pr{.} stands for the probability operator. We say that $\boldsymbol{x} \in D$ is a feasible solution if satisfying the above constraint limits. Introduce the following constraint violation function,

$$\Gamma(\boldsymbol{x})=\sum_{i=1}\max\left\{\alpha_i-\Pr\{G_i(\boldsymbol{x},\eta^i)\le 0\},\ 0\right\}$$
$$+\frac{1}{J}\sum\nolimits_{j=1}^{J}[\max(g_j(\boldsymbol{x}),\ 0)]^2+\frac{1}{K}\sum\nolimits_{k=1}^{K}h_k^2(\boldsymbol{x}), \tag{1}$$

and define

$$\boldsymbol{x}\prec\boldsymbol{y}\Leftrightarrow E[f_i(\boldsymbol{x},\xi)]\le E[f_i(\boldsymbol{y},\xi)]$$
$$\wedge\exists j,s.t.,E[f_j(\boldsymbol{x},\xi)]<E[f_j(\boldsymbol{y},\xi)]. \tag{2}$$

Obviously, if $\Gamma(\boldsymbol{x})=0$, $\boldsymbol{x}$ is feasible. We give the following concept of constraint dominance.

Definition 1 ([14]): For given $\boldsymbol{x},\boldsymbol{y} \in D$, $\boldsymbol{x}$ is said to constrain- dominate $\boldsymbol{y}$, i.e., $\boldsymbol{x} \prec_c \boldsymbol{y}$, if one of the following conditions holds

1) $\boldsymbol{x}$ and $\boldsymbol{y}$ are feasible, and $\boldsymbol{x}\prec\boldsymbol{y}$;
2) $\boldsymbol{x}$ is feasible, but $\boldsymbol{y}$ is infeasible;
3) $\boldsymbol{x}$ and $\boldsymbol{y}$ are infeasible, but $\Gamma\left(\boldsymbol{x}\right)<\Gamma\left(\boldsymbol{y}\right)$.

Let $\{\xi^1,\xi^2,\cdots,\xi^l\}$ and $\{\eta^1,\eta^2,\cdots,\eta^s\}$ be the observations of random vectors ξ and η, respectively. By the law of large number, once l and s are sufficiently large, the above MCCP can be replaced by the following sample average approximation model (SAAM),

$$\min_{x\in D}\left(\hat{f}_1(\boldsymbol{x}),\hat{f}_2(\boldsymbol{x})\cdots,\hat{f}_m(\boldsymbol{x})\right)$$
$$s.t.,\begin{cases}\hat{p}_i(\boldsymbol{x})\ge\beta_i,g_j\left(\boldsymbol{x}\right)\le 0,h_k\left(\boldsymbol{x}\right)=0,\\ \hat{f}_q(\boldsymbol{x})=\frac{1}{l}\sum_{r=1}^{l}f_q\left(\boldsymbol{x},\xi^r\right),\\ \hat{p}_i(\boldsymbol{x})\equiv\frac{1}{s}\sum_{s=\mathbf{1}}^{s}\mathrm{I}\left(G_i\left(\boldsymbol{x},\eta^s\right)<0\right),\\ 1\le i\le I,1\le j\le J,1\le k\le K,1\le q\le m,\end{cases}$$

with $0 \le \beta_i < \alpha_i$, where I(.) is an indicator function, namely when '.' is true, it takes 1 and 0 otherwise. However, once l and s are large enough, the computationally expensive cost will occur necessarily. In order to reduce such high computational cost and acquire MCCP's solutions, we in the present work require that l and s be dynamically determined, namely each candidate solution $\boldsymbol{x}$ is attached two dynamical sample sizes of $l(\boldsymbol{x})$ and $s(\boldsymbol{x})$ and accordingly, SAAM can be reformulated by a sample-dependent approximation model (SDAM) below,

$$\min_{x\in D}\left(\hat{f}_1(\boldsymbol{x}),\hat{f}_2(\boldsymbol{x})\cdots,\hat{f}_m(\boldsymbol{x})\right)$$
$$s.t.,\begin{cases}\hat{p}_i(\boldsymbol{x})\ge\beta_i,g_j\left(\boldsymbol{x}\right)\le 0,h_k\left(\boldsymbol{x}\right)=0,\\ \hat{f}_q(\boldsymbol{x})=\dfrac{1}{l(\boldsymbol{x})}\sum\nolimits_{r=1}^{l(\boldsymbol{x})}f_q\left(\boldsymbol{x},\xi^r\right),\\ \hat{p}_i(\boldsymbol{x})\equiv\dfrac{1}{s(\boldsymbol{x})}\sum\nolimits_{r=1}^{s(\boldsymbol{x})}I\left(G_i\left(\boldsymbol{x},\eta^r\right)<0\right),\\ 1\le i\le I,1\le j\le J,1\le k\le K,1\le q\le m.\end{cases}$$

Audibert [37] intensively investigated the sample estimate theory of random variable, and acquired some valuable probabilistic inequalities. When random variables X_1, X_2,... and X_m take values on a non-negative and bounded interval [0, b], they acquired a probabilistic inequality capable of being used in controlling the sampling size of a random variable. In fact, after carefully analyzing the process of proof of the inequality, we notice that it also holds over a more general bounded and closed interval, which can be described below.

Theorem 1 (Bernstein Bound Estimate [37]): Let $X_1,\cdots,X_t$ be i.i.d real random variables on $[a,b]$ with mean μ and $b>0$; $\bar{X}_t$ and σ_t denotes their empirical mean value and variance, respectively. Then, for any given $\rho > 0$ the following inequality holds,

$$\Pr\left\{|\bar{X}_t-\mu|\le\sqrt{\frac{2\rho\sigma_t}{t}}+\frac{3b\rho}{t}\right\}\ge 1-3e^{-\rho}. \tag{3}$$

IV. SAMPLE BOUND AND FEASIBILITY DETECTION

Introduce a finite set X in R^p, and define

$$X_\alpha=\left\{\boldsymbol{x}\in X|p(\boldsymbol{x})\equiv\Pr\left\{G\left(\boldsymbol{x},\eta\right)\le 0\right\}\ge\alpha\right\}, \tag{4}$$

and

$$X_\beta^t=\left\{\boldsymbol{x}\in X|\hat{p}(\boldsymbol{x})\equiv\frac{1}{t}\sum\nolimits_{j=1}^{t}I\left(G\left(\boldsymbol{x},\eta^j\right)\le 0\right)\ge\beta\right\}, \tag{5}$$

with $0<\alpha,\beta\le 1$ and $\beta<\alpha$.

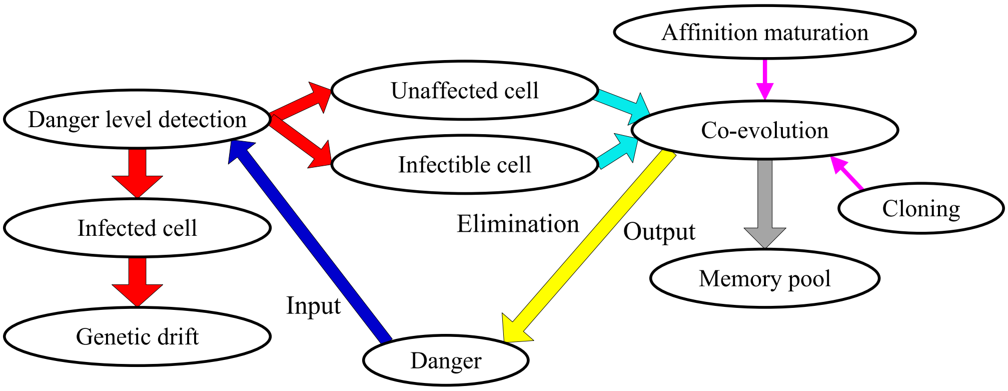


FIGURE 1. Artificial immune optimization model.

Below, we discuss the relation between X_α and X_β^t in terms of Theorem 1, in order to yield a sample bound estimate which X_α includes X_β^t.

Theorem 2: If $0 \leq \beta < \alpha$, then

$$\Pr\left\{X_\beta^t \subseteq X_\alpha\right\} \geq 1 - 3|X|e^{-\rho_0}, \tag{6}$$

where $\rho_0 = \frac{t[(6b(\alpha-\beta)+2\sigma_t]}{18b^2}$.

Proof. Assume that $\boldsymbol{x} \in X_\beta^t \backslash X_\alpha$. Since $p(\boldsymbol{x}) < \alpha$, Eqs.(4) and (5) derive

$$\begin{aligned} &\Pr\left\{\boldsymbol{x} \in X_\beta^t \backslash X_\alpha\right\} \\ &= \Pr\left\{\frac{1}{t}\sum\nolimits_{j=1}^{t} I\left(G\left(\boldsymbol{x}, \eta^j\right) \leq 0\right) \geq \beta, x \notin X_\alpha\right\} \\ &\leq \Pr\left\{\frac{1}{t}\sum\nolimits_{j=1}^{t} I\left(G\left(\boldsymbol{x}, \eta^j\right) \leq 0\right) - E\left[I\left(G\left(\boldsymbol{x}, \eta\right) \leq 0\right)\right] \right. \\ &\quad \left. > -(\alpha - \beta)\right\}. \end{aligned} \tag{7}$$

Take a positive ρ_0 satisfying

$$\sqrt{\frac{2\rho_0\sigma_t}{t}} + \frac{3b\rho_0}{t} = \alpha - \beta. \tag{8}$$

Hence, Theorem 1 and Eq.(7) imply the following inequality by defining $\rho = \rho_0$,

$$\Pr\left\{\boldsymbol{x} \in X_\beta^t \backslash X_\alpha\right\} \leq 3e^{-\rho_0}. \tag{9}$$

Accordingly, Eq.(9) follows

$$\begin{aligned} 1 - \Pr\left\{X_\beta^t \subseteq X_\alpha\right\} &= \Pr\left\{\exists \boldsymbol{x} \in X_\beta^t, s.t., \boldsymbol{x} \notin X_\alpha\right\} \\ &\leq \sum_{\boldsymbol{x} \in X \backslash X_\alpha} \Pr\left\{\boldsymbol{x} \in X_\beta^t\right\} \leq 3|X|e^{-\rho_0}. \end{aligned} \tag{10}$$

Consequently, the above conclusion is true.

Remark 1: The above theorem derives a lower bound estimate model to be used in estimating the probability of each chance constraint in MCCP. More precisely, under a given significance level δ if

$$t \geq \bar{M} \equiv \frac{1}{\rho_0} \ln \frac{3|X|}{\delta}, \tag{11}$$

then $\Pr\{X_\beta^t \subseteq X_\alpha\} \geq 1 - \delta$.

Remark 2: Under given confidences α_i, β_i with $\beta_i < \alpha_i$, let $\bar{M}^i$ denote the lower bound estimate of the i-th chance constraint in MCCP acquired by Eq.(11). If

$$t \geq M \equiv \max\{\bar{M}^i, 1 \leq i \leq I\}, \tag{12}$$

then $\Pr\{X_\beta^t \subseteq X_\alpha\} \geq 1 - \delta$ for all the given chance constraints in MCCP.

Based on the above lower bound estimate on sample size in Eq.(12), a feasibility detection approach in the above algorithm 1 is designed to check whether a candidate solution $\boldsymbol{x}$ is empirically feasible.

Through the above algorithmic formulation, the computational complexity of Algorithm 1 is decided by T_k, where m is updated gradually, where $T_k = m_0 + (k+1)\Delta$, and k is the maximal positive integer satisfying $T_k \leq M$. Additionally, for two candidates $\boldsymbol{x}$ and $\boldsymbol{y}$ in X we say that $\boldsymbol{x}$ empirically dominates $\boldsymbol{y}$ or that $\boldsymbol{y}$ is empirically dominated by $\boldsymbol{x}$ if one of cases (a), (b) and (c) in Definition 1 holds in the case where the expected sub-objective values and the probabilities of chance constraints in MCCP for $\boldsymbol{x}$ and $\boldsymbol{y}$ are replaced by their empirical values.

V. DANGER THEORY

The classical immune theory claims that the immune system can recognize non-self and self-bacterial materials. Once one such system includes non-self-materials, it will trigger a specific immune response and destroy them by immune learning. This theory has been stricken by a recently birthed novel immunological theory, Danger Theory (DT) originally developed by Matzinger [38]. DT differs from self-non-self discrimination. She points out that an immune response is triggered only when some organs or cells are destroyed by *some* harmful bacteria or apoptotic cells called danger. Namely, the immune system only reacts to the danger. She thinks that some auto-reactive processes in the immune system are useful and there must be discrimination happening that goes beyond the self-non-self distinction. She also develops a danger theory model by inserting a new signal called signal zero and the danger into the extended two signal

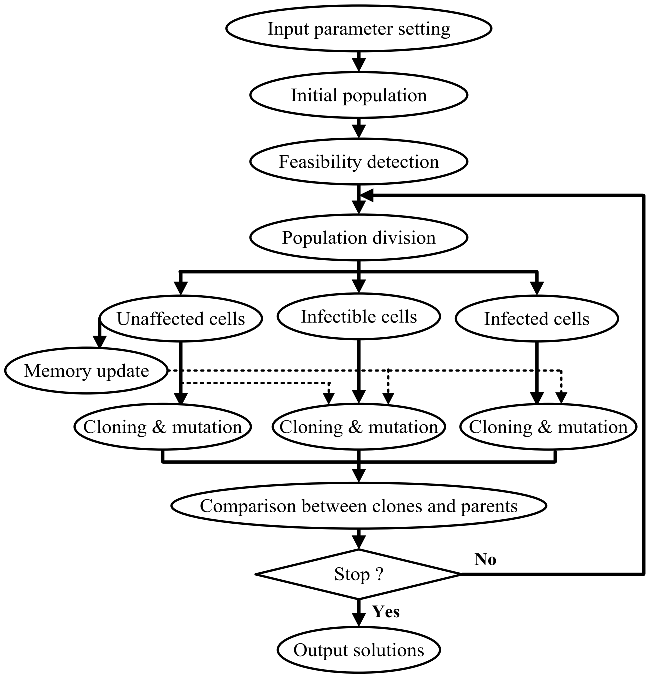


FIGURE 2. The flowchart of ARSMIOA.

model. In one such model, when some immune cells take place abnormal apoptosis because of infection or pathological factors, they activate APCs to create signal two through discharging signal zero (i.e., endogenic danger signals). Such a type of signal helps T cells react to the danger, and hence the adaptive immune response is triggered. It is pointed out that those bacteria, which does not make immune cells abnormally die, cannot urge the immune system to elicit any immune response.

Through the danger theory, the root of which an immune response occurs consists in that there must be some infected immune cells to have harmed one or more cells or organs in bodies. Therefore, we divide immune cells into three broader classes- infected, unaffected and infectible cells. Usually, an infected cell can produce a danger region which makes those cells around it be infected. It is worth mentioning that this theory has become popular. Although there exist different opinions amongst immunologists, such theory leads to a wide way in studying artificial immune systems. Here, an artificial immune optimization model in Fig.1 above is developed to help us solve MCCP in Sect.III.

Fig.1 displays an artificial immune model based on the danger theory, including four functional modules- danger level detection, co-evolution, genetic drift and memory pool. The first is to divide a given cell population into three sub-populations of infected, infectible and unaffected cells. The second makes those unaffected cells gradually become better by cloning and affinition maturation, whereas those infectible cells change their pattern structures by means of memory cells and unaffected ones. The third admits those infected cells to take place genetic drift so that they will be gradually transformed into new unaffected cells. The last collects the best unaffected cells and updates the memory pool.

VI. ALGORITHMIC PRINCIPLE AND STATEMENT

As related to SDAM's optimization model as in Sect.III, the danger is viewed as SDAM itself. For a given finite set X composed of real-coded candidate solutions in D, unaffected cells are regarded as those best candidate solutions in X, namely they are not empirically dominated by other candidate solutions in X through Definition 1 in Sect.III; infectible cells are taken as those candidate solutions in X only empirically dominated by some unaffected cells; infected cells are treated as those worst cells among which each is empirically dominated by some infectible cell; memory cells are taken for the best unaffected cells found until now, i.e., best solutions acquired. Based on these formulations and Fig.1, ARSMIOA's flowchart can be found in Fig.2 above. In this flowchart, Algorithm 1 is first used to check the empirical feasibility of cells in the current population. Once there exist empirically feasible cells, ARRA, proposed by us [39] is cited to compute their empirical objective vectors. Thereafter, the population is divided into three sub-populations of unaffected, infectible and infected cells, each of which executes evolution through cloning and mutation. Finally, a new population is created by non-dominated sorting and comparison. ARSMIOA can be formulated in Algorithm 2.

In algorithm 2, Step 2 is to decide the maximal sampling size which limits the bound of sample size of random variable. Step 4 estimates the amount of constraint violation of each cell by Algorithm 1, by which those empirically feasible cells found are required to further compute their empirical objective values and ranks by ARRA [39]. Steps 6 to 8 divide the current population into three sub-populations with different importance, among which the best sub-population is required to update the memory pool; these sub-populations with different evolutionary fashions urge their elements to discover those high-quality and diverse cells though cloning and mutation, by which a new population is created after comparison between cells. The main modules are designed below.

(a) Evaluation (A_n, M). This computes the probability estimate of each chance constraint for each cell in A_n, depending on Algorithm 1 with the maximal sample size of M. Once some cells in A_n are empirically feasible, they are required to further compute their empirical objective vectors by ARRA with the sample size M.

(b) Division (A_n). This divides A_n into three sub-populations, i.e., B_n, C_n and D_n. More precisely, based on the constraint violations of cells in A_n, those empirical feasible cells constitute A_{n1}, and others form another sub-population A_{n2}. Subsequently, A_{n1} is segmented into three sub-populations by the conventional non-dominated sorting approach [40], i.e., B_n, C_n and $A_{n1}\backslash\{B_n \cup C_n\}$; B_n and C_n only includes those empirically non-dominated cells in A_{n1} and $A_{n1}\backslash B_n$, respectively. D_n consists of three sub-populations D_{n1}, D_{n2} and D_{n3}, where D_{n1} is formed by those cells in

Algorithm 1 Feasibility Detection Approach

```
Step 1. Input: Candidate solution $\boldsymbol{x}$, all the constraint conditions in MCCP, initial sample size $m_0$,
        maximal sample size $M$ in Eq.(12), and sampling increment $\Delta$;
Step 2. $i \leftarrow 1$;
Step 3. While $i \leq I$ do
    Step 3.1. $t \leftarrow m_0, \sigma_t \leftarrow 0, D \leftarrow 0$;
    Step 3.2. Under sample size $t$, compute $\hat{p}_i(x)$ in SDAM, and set $D \leftarrow \hat{p}_i(\boldsymbol{x})$;
    Step 3.3. If $t > M$ or $\beta_i - \hat{p}_i(\boldsymbol{x}) > \sqrt{\frac{2\rho_0\sigma_t}{t}} + \frac{3b\rho_0}{t}$, then return Step 3.8, and Step 3.4 otherwise;
    Step 3.4. According to sample size increment $\Delta$, calculate
              $\hat{p}'_i(\boldsymbol{x}) \equiv \frac{1}{\Delta}\sum_{r=1}^{\Delta} I(G_i(\boldsymbol{x}, \eta^r) < 0)$;
    Step 3.5. $\hat{p}_i(\boldsymbol{x}) \leftarrow \left(\hat{p}_i(\boldsymbol{x}) \times t + \hat{p}'_i(\boldsymbol{x})\right) / (t + \Delta), D \leftarrow \hat{p}(\boldsymbol{x}) - D$
    Step 3.6. $\sigma_t \leftarrow [(\hat{p}_i(\boldsymbol{x}) - \hat{p}'_i(\boldsymbol{x}))^2 + i\sigma_t + iD^2]/i$;
    Step 3.7. $t \leftarrow t + \Delta$, return Step 3.2;
    Step 3.8. $i \leftarrow i + 1$, return Step 3;
Step 4. End while.
Step 5. Compute the amount of constraint violation by replacing $p_i(\boldsymbol{x})$ with $\hat{p}_i(\boldsymbol{x})$ in Eq.(1),
        and justify whether $\boldsymbol{x}$ is empirically feasible.
```

Algorithm 2 Adaptive Racing Sampling Multi-Objective Algorithm (ARSMIOA)

```
Step 1. Input population size $N$, initial sampling size $m_0$, memory size $M_{\max}$,
    threshold $\sigma_{th}$, sampling increment $\Delta$, clonal size $c$, maximal iterative number $G_{\max}$;
Step 2. Decide the maximal sampling size $M$ as in Eq.(12);
Step 3. Set $n =: 0$, $M_{\text{set}} := \phi$, and generate $A_n$ composed of $N$ random cells;    // Initialization
Step 4. While $n < G_{\max}$ do
Step 5. Evaluate all elements in $A_n$: Evaluation $(A_n, M)$;    // cell evaluation
Step 6. Execute division: $(B_n, C_n, D_n) :=$ Division$(A_n)$;    // Create sub-populations
Step 7. Update memory pool: $M_{\text{set}} =:$ Update$(M_{\text{set}}, B_n)$;    // Memory cell update
Step 8. Perform evolution:    // Co-evolution
    Step 8.1. $(B_n^*, C_n^*) :=$ Evolution $a(B_n, C_n, M_{\text{set}})$;
    Step 8.2. $D_n^* :=$ Evolution $b(D_n)$;
Step 9. Implement population update:
    $A_{n+1} :=$ Comparison$(B_n \cup B_n^*; C_n \cup C_n^*; D_n \cup D_n^*)$;    // Population update
Step 10. $n =: n + 1$;
Step 11. End while;
Step 12. Output memory cells.
```

$A_{n1} \backslash \{B_n \cup C_n\}$; depending on the version of danger region in Sect.V, D_{n2} includes those empirically infeasible cells in A_{n2} which their constraint violations are smaller than a given threshold σ_{th}; other cells in A_{n2} constitutes D_{n3}. Therefore, $D_n = D_{n1} \cup D_{n2} \cup D_{n3}$.

(c) Update (M_{set}, B_n). This first sends those elements in B_n into M_{set}. Second, those identical and empirically dominated cells in M_{set} are directly eliminated. Thereafter, if $|M_{\text{set}}| > M_{\max}$, those memory cells with small crowding distances [40] are deleted.

(d) Evolution $a(B_n, C_n, M_{\text{set}})$. Each cell $\boldsymbol{x}$ in B_n and C_n proliferates c and (c-1) clones, respectively; the variants of the clones of $\boldsymbol{x}$ in B_n execute local mutation, i.e.,

$$x'_i \leftarrow x_i + \xi(b_i - a_i), 1 \leq i \leq p;$$

with uniformly distributed random variable $\xi \in (-1, 1)$ and mutation rate $p_m(\boldsymbol{x}) = (1 + e^{r(\boldsymbol{x})})^{-1}$, where a_i and b_i are the lower and upper bounds of x_i, and $r(\boldsymbol{x})$ is the rank of $\boldsymbol{x}$ acquired by ARRA. Each clone of $\boldsymbol{x}$ in C_n implements the following polynomial-like mutation with $p_m(\boldsymbol{x})$,

$$\boldsymbol{x}' \leftarrow \boldsymbol{x}_M + d(\boldsymbol{z} - \boldsymbol{x})_, d = \begin{cases} (2\varsigma)^{\frac{1}{1+\lambda}} - 1, & \eta < 0.5, \\ 1 - (2(1-\varsigma))^{\frac{1}{1+\lambda}}, & else, \end{cases}$$

with uniformly distributed random variables $\varsigma, \eta \in (0, 1)$; $\boldsymbol{x}_M$ and $\boldsymbol{z}$ is randomly picked up in M_{set} and $\mathrm{B}_n \cup \mathrm{C}_n$, respectively. The mutated clones of cells in B_n and C_n constitute B_n^* and C_n^*, respectively. Here, λ denotes the variation amplitude, usually taking 5.

(e) Evolution $b(D_n)$. Recall $D_n = D_{n1} \cup D_{n2} \cup D_{n3}$ above. Each cell $\boldsymbol{x}$ in D_{n1} produces (c-1) clones, but each element in D_{n2} and D_{n3} only creates a clone. Each element $\boldsymbol{x}$ in D_{n1} with mutation rate 1 performs the Gaussian mutation, but that in D_{n2} mutates by the following fashion,

$$\boldsymbol{x}' \leftarrow \xi(\boldsymbol{x}_M - \boldsymbol{x}).$$

Additionally, the clones of elements in D_{n3} implement genetic drift with mutation rate 0.5, i.e.,

$$(x_1 \ldots, x_i, \ldots, x_p) \leftarrow (x_1 \ldots, x_i', \ldots, x_p).$$

Thereafter, all the mutated clones constitute D_n^*.

(f) Comparison ($B_n \cup B_n^*$; $C_n \cup C_n^*$; $D_n \cup D_n^*$). This is to acquire a new population of A_{n+1}. To this end, let $E_n = B_n^* \cup C_n^* \cup D_n^*$, and $cl(\boldsymbol{x})$ denotes the set of mutated clones of $\boldsymbol{x}$ in A_n. Firstly, all elements in E_n need to carry out evaluation by Step 4. Secondly, all clones in B_n^*, C_n^* and D_n^* participate in competition with their parents so as to acquire A_{n+1}. In other words, for each given cell $\boldsymbol{x}$ in A_n if its best clone $\boldsymbol{y}$, which is not empirically dominated by other mutated clones in $cl(\boldsymbol{x})$, empirically dominates $\boldsymbol{x}$, A_{n+1} includes $\boldsymbol{y}$, and $\boldsymbol{x}$ enters A_{n+1} otherwise.

VII. COMPUTATIONAL ANALYSIS

A. COMPUTATIONAL COMPLEXITY

ARSMIOA's computational complexity is decided by Steps 5, 7, 8 and 9, for which the conclusion is given below.

Theorem 3: Assume that I, J and K in Sect.III are small. Then, in the worst case ARSMIOA's computational complexity is with

$$O((N + M_{\max}) \log(N + M_{\max}) \\ + N(cp + M_2 + c \log cN + IT_k)).$$

Proof. For each element in the current population, Step 5 needs to estimate the probability and the amount of constraint violation for each chance constraint by Algorithm 1 with $O(N(IT_k{+}J{+}K))$. In addition, since ARRA is with $O(N(M_2 + \log N))$, its complexity is with

$$O(N(M_2 + \log N + IT_k + J + K)),$$

where

$$M_2 \equiv \left(\frac{\sqrt{2}(b-a)}{0.05}\right)^2 \ln \frac{2N}{\delta}, \tag{13}$$

and a and b denote the lower and upper bounds of the stochastic sub-objectives in MCCP in Sect.III. Since the size of the memory pool is below $M_{\max}$, Step 7 is with $O((N + M_{\max}) \times \log(N + M_{\max}))$. Step 8 enforces mutation with at most cN times, as the size of its clones is not beyond cN. Consequently, its computational complexity is $O(cpN)$. Further, Step 9 is with $O(cN \log cN)$. Summarily, since I, J and K are small, ARSMIOA's computational complexity in the worst case can be decided by

$$\begin{aligned} O_c &= O(N(IT_k + J + K)) + O(N(M_2 + \log N)) \\ &\quad + O((N + M_{\max}) \log(N + M_{\max})) \\ &\quad + O(cpN) + O(cN \log cN) \\ &= O((N + M_{\max}) \log(N + M_{\max}) + N(cp + M_2 \\ &\quad + c \log cN + IT_k)). \end{aligned}$$

Through the above theorem, the complexity of ARSMIOA is determined by $T_k, N, M_2, M_{\max}, c$ and p. However, since $T_k \leq M$, Eqs.(11) to (13) indicate that the complexity is decided mainly by N, p, c and $M_{\max}$. Once N and c are small enough, ARSMIOA will be with low computational complexity. Thus, we in the current work require that N and c be small as possible.

B. PERFORMANCE CRITERIA

Three popular criteria [4] are recalled to execute comparison between two multi-objective algorithms A and B. Assume that such two algorithms solve some optimization problem with one times, and correspondingly they acquire empirically non-dominated sets P and Q in order.

A1. Coverage rate (CR). This can measure the difference of the qualities of solution sets P and Q by means of the version of constraint dominance as in Sect. III, given by

$$CR(A, B) = \frac{|\{\boldsymbol{x} \in Q | \exists \boldsymbol{y} \in P, s.t., \boldsymbol{y} \prec_{\hat{c}} \boldsymbol{x}\}|}{|Q|}, \tag{14}$$

where $\boldsymbol{y} \prec_{\hat{c}} \boldsymbol{x}$ denotes that $\boldsymbol{y}$ empirically constrain-dominates $\boldsymbol{x}$ by Definition 1. Obviously, $0 \leq CR(A, B) \leq 1$. If $CR(A, B) > CR(B, A)$, algorithm A can achieve better solution search than algorithm B. If $CR(A, B) = 1$, the quality of set P is absolutely superior to that of set Q.

A2. Coverage density (CD). This can measure the distributional characteristic of solutions in P, defined by

$$CD = \frac{1}{|P| - 1} \sum_{j=1}^{|P|} (d_j - \bar{d})^2,$$

$$d_j = \min_{j \neq i, 1 \leq k \leq |P|} \left\{ \left\| \boldsymbol{x}_j - \boldsymbol{x}_k \right\|, \mathbf{x}_j, \mathbf{x}_k \in P \right\}, \bar{d} = \frac{1}{|P|} \sum_{j=1}^{|P|} d_j. \tag{15}$$

Eq.(14) indicates that CD takes values within 0 and 1. If CD=0, all elements in P is with uniform distribution, and thus if CD is smaller, the distribution of solutions in P is better.

A3. Coverage scope (CS). This represents the coverage width of solutions in P, given by

$$CS = \max_{1 \leq j,k \leq |P|} \left\{ \left\| \boldsymbol{x}_j - \boldsymbol{x}_k \right\|, \mathbf{x}_j, \mathbf{x}_k \in P \right\}. \tag{16}$$

If CS is large, set P covers a wide scope, and thus set P is better when CS is larger.

VIII. EXPERIMENTAL STUDY

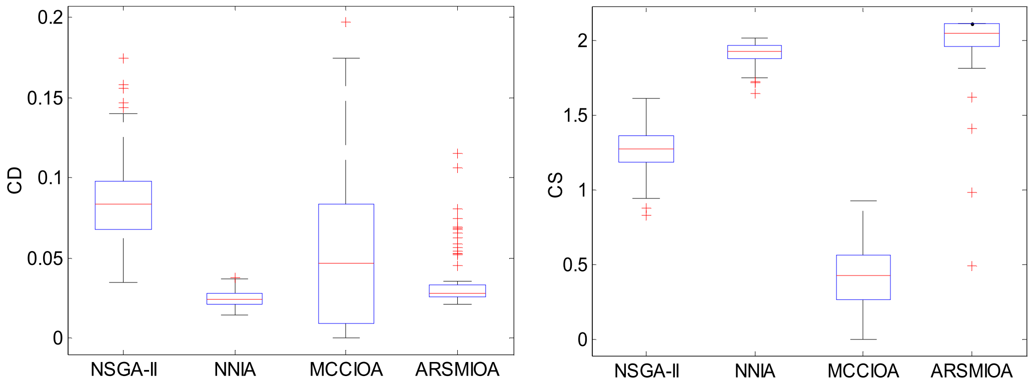
Our experiments are implemented on a computer with CPU/3.50GHz and RBM/3.00 GB by means of Visual C ++ platform. The proposed approach, ARSMIOA as in Sect.VI, is compared against three competitive multi-objective approaches, namely one non-dominated sorting genetic algorithm (NSGA-II [40]) and two multi-objective immune optimization algorithms (MCCIOA [14], NNIA [42]). All the approaches are executed on seven theoretical multi-objective chance constrained benchmark problems in Appendix A and two engineering MCCP problems in Appendix B acquired by modifying the reported static benchmark problems [41]. It is emphasized that, although designed to solve static

TABLE 1. Comparison of statistical results for problems CP1 to CP7; the bold color denotes the best result under a given index for a given problem.

Type	Algorithm	*ACR* (%)				*CD*		*CS*		*FR* (%)	*AR* (*Sec.*)
		NSGA-II	NNIA	MCCIOA	ARSMIOA	*Mean*	*St.Dev*	*Mean*	*St.Dev*		
CP1	NSGA-II	0	0	0.04	0	0.09	0.03	1.27	0.15	73	7.1
	NNIA	100	0	100	7	0.04	0.03	1.91	**0.07**	95	0.8
	MCCIOA	96	0	0	0	0.06	0.05	0.42	0.19	11	2.8
	ARSMIOA	**1**	**84**	**1**	0	**0.03**	**0.02**	**2.00**	0.22	**98**	**0.4**
CP2	NSGA-II	0	4	32	1	0.48	0.59	0.91	1.18	95	5.62
	NNIA	95	0	98	1	0.03	**0.01**	1.53	**0.10**	72	1.77
	MCCIOA	56	0	0	0	0.05	0.30	1.02	0.87	75	2.43
	ARSMIOA	**98**	**92**	**99**	0	**0.03**	0.02	**1.67**	0.15	**100**	**1.57**
CP3	NSGA-II	0	0	0	0	1.86	0.36	34.05	4.90	58	3.96
	NNIA	100	0	38	1	0.51	0.35	22.29	5.18	99	1.53
	MCCIOA	100	43	0	2	3.20	2.76	20.52	8.24	99	2.92
	ARSMIOA	**100**	**65**	**38**	0	**0.51**	**0.17**	**36.36**	**3.52**	**99**	**1.45**
CP4	NSGA-II	0	64	81	2	0.12	0.43	1.17	1.68	6	2.60
	NNIA	36	0	85	4	0.09	**0.31**	1.16	0.99	27	1.50
	MCCIOA	19	15	0	18	0.13	0.46	**1.21**	**0.45**	50	2.78
	ARSMIOA	**98**	**96**	**82**	0	**0.09**	0.51	1.20	0.74	**88**	**1.44**
CP5	NSGA-II	0	50	48	0	0.08	0.65	1.02	1.95	71	2.25
	NNIA	50	0	66	1	0.23	1.27	0.03	0.76	77	1.59
	MCCIOA	38	14	0	0	0.22	1.93	**4.76**	2.68	94	2.81
	ARSMIOA	**97**	**90**	**99**	0	**0.07**	**0.49**	1.09	**0.45**	**100**	**1.35**
CP6	NSGA-II	0	6	86	0	0.02	0.02	1.00	0.29	22	3.56
	NNIA	94	0	98	38	0.05	0.02	0.88	0.22	79	3.06
	MCCIOA	14	2	0	4	0.04	**0.02**	0.02	**0.11**	62	3.66
	ARSMIOA	**1**	**51**	**95**	0	**0.02**	0.03	**1.18**	0.17	**94**	**2.63**
CP7	NSGA-II	0	59	68	0	0.03	**0.05**	2.61	0.74	95	3.44
	NNIA	17	0	93	0	0.12	0.11	1.51	0.79	98	3.16
	MCCIOA	5	3	0	0	**0.02**	0.06	2.66	0.67	89	4.16
	ARSMIOA	**88**	**94**	**100**	0	0.04	0.06	**3.38**	**0.46**	**100**	**3.17**

ACR(*A*,*B*) denotes the average of coverage rates *CR*(*A*,*B*) obtained by algorithm *A* via algorithm *B*; *FR* stands for the rate of feasible solutions among all solutions acquired after 100 runs; *AR* represents the average runtime of a given algorithm.

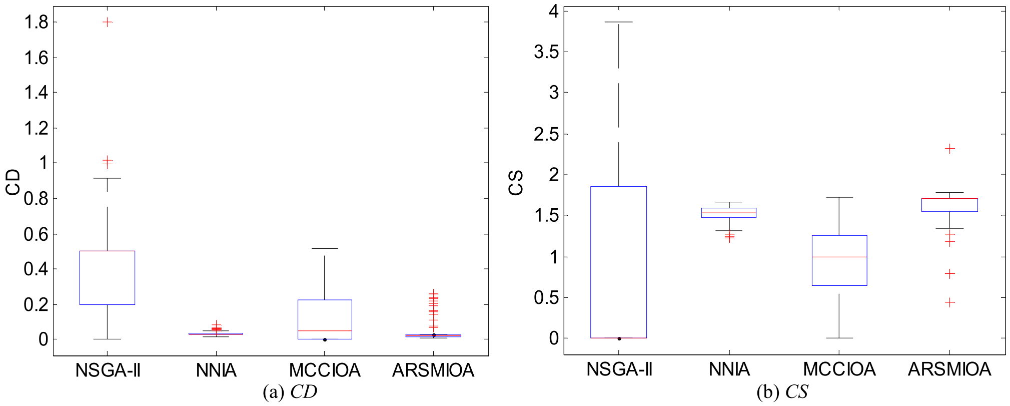
multi- objective programming problems, NSGA-II and NNIA are competitive for MCCP problems when each individual is attached the same large sample size; MCCIOA is an optimizer with the strategy of adaptive sampling, specially designed for MCCP problems. We also emphasize that, apart from MCCIOA, no other algorithmic achievements with adaptive sampling have been found for nonlinear MCCP. Among the three compared algorithms, by experimental trails NNIA and NSGA-II are with the same population size 100 and also the same sample size 300 for each individual; NNIA's crossover rate and memory size are 0.1 and 100, respectively; NSGA-II's crossover and mutation rates are 0.6 and 0.1 in order. MCCIOA takes population size 60 and memory size 50. Their other parameter settings can be found in their literatures. In order to ensure the fairness of comparison, all the approaches terminate their solution search procedures when the total of their respective evaluations is 5×10^6, while each approach executes 100 single runs on each test example.



(a) *CD*

(b) *CS*

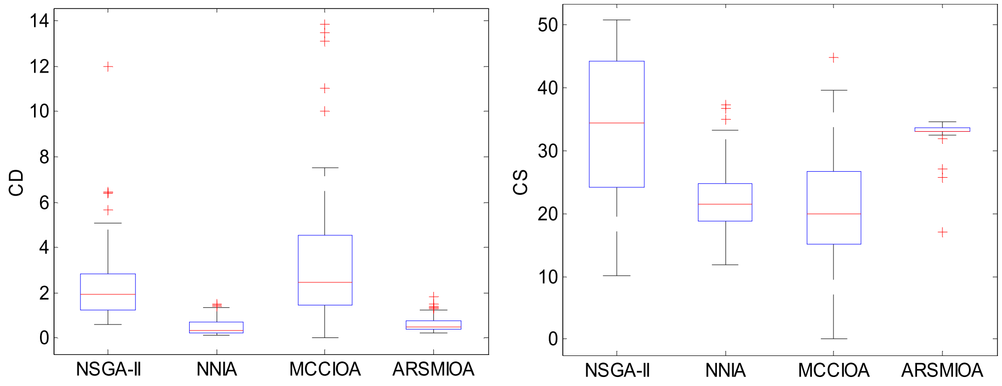
FIGURE 3. Comparison of box plots of values on *CD* and *CS* for e problem CP1.



(a) *CD*

(b) *CS*

FIGURE 4. Comparison of box plots of values on *CD* and *CS* for e problem CP2.



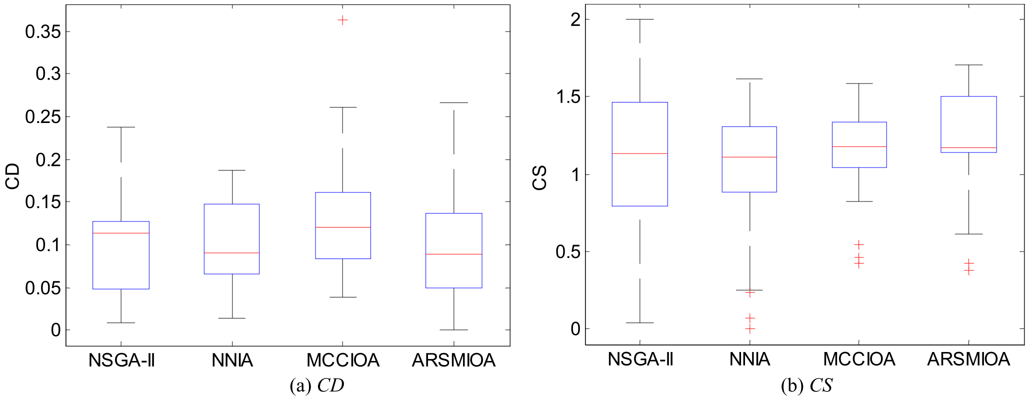
(a) *CD*

(b) *CS*

FIGURE 5. Comparison of box plots of values on *CD* and *CS* for e problem CP3.

A. PARAMETER SETTING

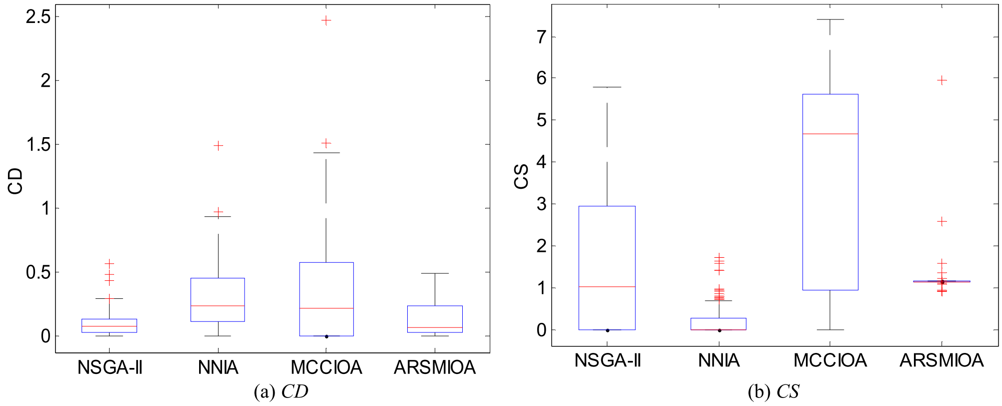
As mentioned in Algorithm 2, ARSMIOA includes five parameters, i.e., N, m_0, $\sigma_{\mathrm{th}}, M_{\max}$ and Δ. Theorem 3 shows that N, $M_{\max}$ decides the computational complexity of ARSMIOA. if they take large values, ARSMIOA can ultimately acquire many solutions, and meanwhile its performance characteristics can be more clearly found. However, in such case, its computational cost is large. Thus, N take values within 5 and 10, and $M_{\max}$ does so within 60 and 100.Additionally, m_0 as an initial sample size influences the quality of solution search, generally changing within 40 and 50. σ_{th} as a danger radius divides an empirically infeasible population into two sub-populations, usually taking values within 0.05 and 0.2. c as a clonal size influences the ability of individual exploitation and the speed of solution search, taking values within 3 and 10. Δ, presented in Algorithm 1 is a sampling increment to influence the speed of feasibility detection; its value range is usually within 10 and 15. Additionally, α_i and β_i, displayed in MCCP and SDAM in Sect.III respectively are the confidence levels, changing



(a) *CD*

(b) *CS*

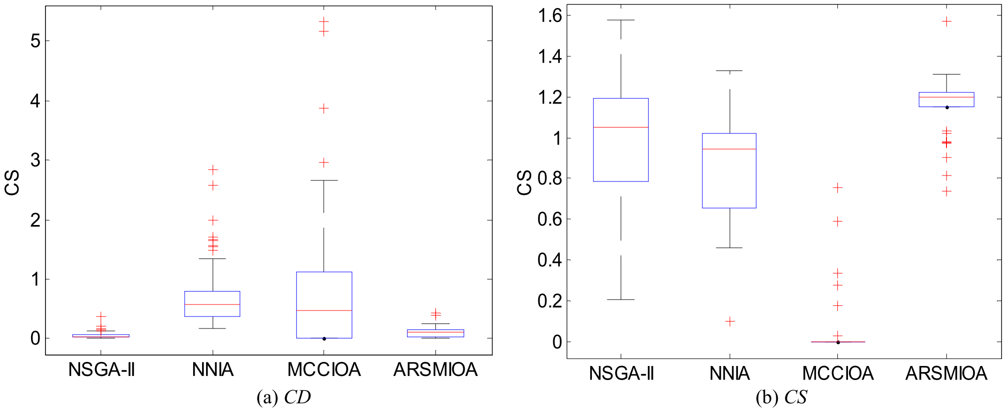
FIGURE 6. **Comparison of box plots of values on *CD* and *CS* for e problem CP4.**



(a) *CD*

(b) *CS*

FIGURE 7. **Comparison of box plots of values on *CD* and *CS* for e problem CP5.**



(a) *CD*

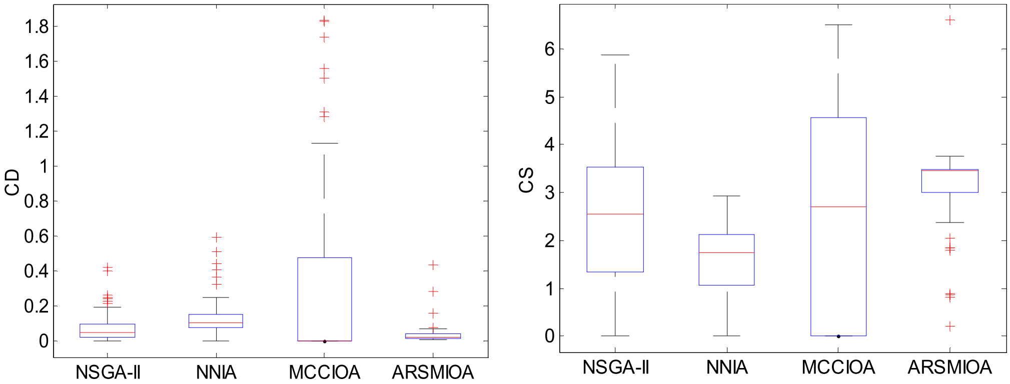
(b) *CS*

FIGURE 8. **Comparison of box plots of values on *CD* and *CS* for e problem CP6.**

within 0.8 and 1. By experimental tuning, we define N=10, $m_0 = 30$, $\Delta = 10$, $c = 3$, $M_{\max} = 100$, $\sigma_{th} = 0.1$, $\alpha_i = 0.2$ and $\beta_i = 0.19$ with $1 \leq i \leq I$.

B. PROBLEM DESIGN AND ANALYSIS

Zhang et al. [41] developed a suite of static multi-objective programming problems including 13 non- constrained multi-objective programming problems UF1 to UF13 and 10 constrained ones CF1 to CF10. In order to sufficiently examine the performance characteristics of ARSMIOA, we here modify seven of them (i.e., CP1 to CP7) into MCP problems, by respectively adding a Gaussian noise with the standard normal distribution to their respective sub-objective functions and transforming their constraints into chance constraints (See Appendix A). Such MCCP problems are still named by the same versions as their original ones. It is worth pointing out that the original problems CP1 to CP7 with strong non-linearity or multi-modality include respectively two sub-objective functions which link up only by variable x1.



(a) *CD*

(b) *CS*

FIGURE 9. Comparison of box plots of values on *CD* and *CS* for e problem CP7.

Among these problems, CP1 has a Pareto front composed of (2*n*+1) discrete points; CP2 and CP3 involve in two disconnected Pareto fronts, each of which is formed by discrete points. Other test problems CP4 to CP7 involve in piece-wise sub-objective functions, and meanwhile their Pareto fronts consist of extremely complex piece-wise segments. Thereafter, each of the above four approaches solves each of the nine problems with 100 times, while the related experimental results are used to compare the performance characteristics of the algorithms.

C. EXPERIMENTAL ANALYSIS

Based on the three test criteria, the above four approaches acquire their statistical results given in Table 1. Correspondingly, the distributional characteristics of their solutions are formulated by the box plots in Figs.3 to 6 above and 7 to 9 below.

Table 1 indicates that there exist some distinct differences between the above approaches with the aspects of solution quality, solution search stability, solution distribution, coverage scope and so forth. We can draw the following conclusions.

Comparison on Solution Quality: The values on *FR* illustrates the above four algorithms have different abilities of exploiting those regions at which feasible solutions exist. ARSMIOA can find feasible solutions with a higher probability than each compared approach for each problem. It can acquire feasible solutions with probability 100% for CP2, CP5 and CP7, 98∼99% for CP1 and CP3, and 88∼94% for CP4 and CP6. This hints that its adaptive sampling approaches on objective and constraint handling can effectively decide the importance of a cell in the current population, namely such sampling approaches enable the estimates of objective and constraints of each cell to approach the related theoretical values with increasing iterative number. Its multiple mutation strategies help those existing cells move towards the feasible region(s) as possible. We also notice that by comparison, the three compared approaches can acquire their feasible solutions with relatively high probabilities for some test problems but are opposite for others. We emphasize a fact that, since any feasible solution can dominate any unfeasible solutions according to Definition 1 as in Sect. III, a feasible solution set always covers any infeasible solution set. Based on this point, those average coverage rates in columns 3 to 6 in Table 1 expose that the above four approaches have significantly different capabilities of population evolution and noise handling. ARSMIOA can clearly acquire the best solution qualities for the above examples, since there are always the following average coverage relationships for any given test example:

TABLE 2. Comparison of values on *ACD*, *ACS* and *FR* under different parameter settings.

(m_0, Δ)	*ACD*	*ACS*	*FR* (%)	(m_0, Δ)	*ACD*	*ACS*	*FR* (%)
(10, 5)	0.03	1.91	99	(60, 5)	0.03	1.87	99
(20, 10)	0.04	1.72	98	(50, 10)	0.03	1.75	99
(30, 15)	0.03	1.90	98	(40, 15)	0.02	1.80	98
(40, 20)	0.04	1.75	99	(30, 20)	0.02	1.65	99
(50, 25)	0.04	1.92	98	(20, 25)	0.03	1.91	99
(60, 30)	0.05	2.03	99	(10, 30)	0.04	1.73	98

$$\begin{aligned} &ACR(\text{ARSMIOA, NSGA-II}) \\ &\quad \geq ACR(\text{NSGA-II, ARSMIOA}), \\ &ACR(\text{ARSMIOA, NSGA-II NNIA}) \\ &\quad \geq ACR(\text{NNIA, ARSMIOA}), \\ &ACR(\text{ARSMIOA, MCCIOA}) \\ &\quad \geq ACR(\text{MCCIOA, ARSMIOA}). \end{aligned}$$

The above inequalities can sufficiently illustrate that ARSMIOA is superior to other approaches with the aspect of solution quality when solving each test problem, for which one of the main reasons consists in that Algorithms 1 and ARRA can effectively suppress noisy interference when discriminating those high-quality cells from those poor ones in the current population. For example, for CP7 those non-dominated sets, found by ARSMIOA cover averagely in order 88%, 94% and 100% of those obtained by NSGA-II, NNIA and MCCIOA. Conversely, the latter three approaches can only get smaller *ACR* values by comparison with ARSMIOA. On the other hand, NSGA-II can acquire a better solution quality than either NNIA or MCCIOA when handling CP 4, CP 5 or CP7, but behaves worse for CP1, CP2 and CP3. When solving CP6, it is superior to MCCIOA only with the aspect of search effect; NNIA performs well over NSGA-II for CP 1, CP 2, CP 3 and CP6 as well as MCCIOA for all the test problems but CP3; MCCIOA outperforms NSGA-II for CP 1 and CP 2 as well as NNIA for CP3. Summarily, we can assert that with regard to solution quality, ARSMIOA behaves best; NNIA is secondary, whereas MICCIOA performs poor.

Distribution and Coverage Scope: The values on *Mean* in *CD* given in columns 7 and 8 in Table 1 can draw a conclusion that the above four approaches have distinct solution distribution characteristics. By comparison against the compared approaches, ARSMIOA can find some solutions with relatively uniform distributions for all the test examples, due to its small means and variances in the table. NNIA can also obtain better solution distributions than NSGA-II and MCCIOA for all the test problems except for CP6 and CP7. The latter two can only get satisfactory solution distributions for a very few test problems. On the other hand, the values on *St.Dev* in *CD* indicate that ARSMIOA can always acquire relatively stable solution distribution for any given test problem in each execution, whereas the compared approaches cause instable solution distribution for some of the test problems. Figs.3 (a) to 9(a) show that the solution distribution of ARSMIOA is relatively uniform for each test problem and can keep stable for each run. Such figures also hint that MCCIOA presents distinctly instable solution search performance while the phenomenon of fluctuation of its solution quality is severe. Additionally, NSGA-II also causes instable solution distribution for CP 1, CP2, CP3 and CP6, while NNIA does so for CP6 and CP7.

The values on *CS*, displayed in columns 9 and 10 derive that all the solutions, acquired by ARSMIOA in each run can widely spread over the relatively stable non-dominated fronts for all the test examples except for CP5. This indicates that ARSMIOA has the capabilities of strong diversity and population exploitation. It is pointed out that, whereas ARSMIOA can only find some solutions with a relatively narrow average coverage scope by comparison to MCCIOA when coping with CP5, MCCIOA causes a larger variance on CS and thus its solution coverage scope is instable. Additionally, we observe that NNIA and MCCIOA need to make further improvements on their diversity of population, since their solution sets acquired for CP1, CP5 or CP6 only cover very narrow scopes and hence they get easily into local search. NSGA-II can widely perform solution search because of its strong exploration. Figs.3 (b) to 9(b) also hint that NSGA-II and MCCIOA present instable solution search on solution coverage scope. ARSMIOA's solution coverage scope keeps relatively stable for each test problem above. NNIA can obtain stable but relatively narrow solution coverage scopes for the test problems.

D. SENSITIVITY ANALYSIS

ARSMIOA includes two crucial parameters (m_0, Δ) which directly influence its search effect and efficiency. m_0 is the initial sampling size of handling each chance constraint in MCCP, while Δ is the sampling increment. Take CP1 for example. Under different combinations of values on m_0 and Δ, ARSMIOA's statistical results are given in Table 2.

Table 2 illustrates that the settings of the two parameters slightly influence ARSMIOA's solution quality. When m_0 is directly proportional to Δ, the values on *ACD* and *ACS*, acquired by ARSMIOA increase totally; in other words, when m_0 and Δ are large, ARSMIOA's solution distribution becomes poor but the solution's coverage scope does wide. When m_0 is conversely proportional to Δ, the distribution density and coverage scope of ARSMIOA's solution set degrade gradually, namely when m_0 decreases and Δ increases, ARSMIOA's solution quality becomes poor. Summarily, ARSMIOA is not sensitive to the settings of m_0 and Δ, provided that $40 \leq m_0 \leq 60$ and $5 \leq \Delta \leq 15$.However, in order to take a trade-off between effect and efficiency, the two parameters should take values with $40 \leq m_0 \leq 50$ and $10 \leq \Delta \leq 15$.

IX. CONCLUSION

This work concentrates on probing into an adaptive racing sampling-based multi-objective immune optimization approach (ARSMIOA) inspired by the danger theory, solving a challenging and provoking topic in the context of stochastic programming- nonlinear multi-objective chance constrained programming with unknown noise distribution. We first transform one such kind of multi-objective programming into a sample-dependent multi-objective chance constrained programming model which the sampling size of random variable depends on the quality of each candidate solution. For a given finite set in the decision space, the relation of inclusion between theoretically and empirically feasible solution subsets is studied, and correspondingly a useful sample bound estimate on sample size is derived, based on a mean-valued estimate theorem. Once a candidate solution is required to detect whether to be empirically feasible, one such estimate can be used to control its maximal sampling size, which can reduce ARSMIOA's computational complexity. To effectively execute individual evaluation, a reported racing-based objective evaluation algorithm (ARRA) with adaptive sampling is adopted to compute the empirical objective values of individuals included in a given population; a new feasibility detection approach is designed to efficiently justify whether an individual is empirically feasible, by which the probability estimate of a chance constraint can be acquired. Thereafter, as related to the danger theory in immunology, some bio-immune inspirations are borrowed to develop an artificial immune optimization mechanism and accordingly, ARSMIOA is developed to solve MCCP problems. The theoretical analysis demonstrates that ARSMIOA's computational cost is decided mainly by N, p and $M_{\max}$. By seven hard MCCP problems, comparative experiments have illustrated that ARSMIOA is a competitive optimizer and also performs well over the compared approaches for the nonlinear multi-objective chance constrained problems without prior noise information. The sensitivity analysis has indicated that its crucial parameters only slightly influence its robustness. Additionally, whereas we do our best to study how to explore an immune optimization approach for MCCP, some issues still need to be further studied. For example, its structures need to be optimized in the precondition of improving its solution quality, while the stability of its solution search for the above engineering problems needs to be further advanced.

APPENDIX

A. CP1

$$
\min \ (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))
$$

$$
s.t., \begin{cases}
f_1(\boldsymbol{x}) = x_1 + \dfrac{2}{|J_1|}\displaystyle\sum_{j\in J_1}\left(x_j - x_1^{0.5\left(1.0+\frac{3(j-2)}{n-2}\right)}\right)^2, \\
f_2(\boldsymbol{x}) = 1 - x_1 + \dfrac{2}{|J_2|}\displaystyle\sum_{j\in J_2}\left(x_j - x_1^{0.5\left(1.0+\frac{3(j-2)}{n-2}\right)}\right)^2, \\
J_1 = \{j \,|j \textit{ is odd and } 2 \le j \le n\}, \\
J_2 = \{j \,|j \textit{ is even and } 2 \le j \le n\}, \\
\Pr\{(f_1 + f_2 - a\,|\sin N\pi\ (f_1 - f_2 + 1)| - \xi \ge 0\} \ge 0.8, \\
\xi \sim U(1.2, 1.8), N = 10, a = 1, \\
\quad n = 10, x_1, \ldots, x_n \in [0, 1].
\end{cases}
$$

B. CP2

$$
\min \ (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))
$$

$$
\begin{aligned}
s.t., \ & f_1(\boldsymbol{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right)\right)^2, \\
& f_2(\boldsymbol{x}) = 1 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \cos\left(6\pi x_1 + \frac{j\pi}{n}\right)\right)^2, \\
& J_1 = \{j \,|j \textit{ is odd and } 2 \le j \le n\}, \\
& J_2 = \{j \,|j \textit{ is even and } 2 \le j \le n\}, \\
& \Pr\left\{\frac{t}{1 + e^{4|t|}} - \xi \ge 0\right\} \ge 0.8, \\
& t = f_2 + \sqrt{f_1} - a\sin\left[N\pi\left(\sqrt{f_1} - f_2 + 1\right)\right] - 1, \\
& \xi \sim N(1, 0.5), x_1 \in [0, 1], x_2, \ldots, x_n \in [-1, 1], \\
& N = 2, a = 1, n = 10.
\end{aligned}
$$

C. CP3

$$
\min \ (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))
$$

$$
s.t., \begin{cases}
f_1(\boldsymbol{x}) = x_1 + \dfrac{2}{|J_1|}\left(4\displaystyle\sum_{j\in J_1} y_j^2 - 2\prod_{j\in J_1}\cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 2\right), \\
f_2(\boldsymbol{x}) = 1 - x_1^2 + \dfrac{2}{|J_2|} \\
\left(4\displaystyle\sum_{j\in J_2} y_j^2 - 2\prod_{j\in J_2}\cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 2\right), \\
J_1 = \{j \,|j \textit{ is odd and } 2 \le j \le n\}, \\
J_2 = \{j \,|j \textit{ is even and } 2 \le j \le n\}, \\
y_j = x_j - \sin\left(6\pi x_1 + \dfrac{j\pi}{n}\right), j = 2, \ldots, n, \\
\Pr\left\{f_2 + f_1^2 - a\sin N\pi\left(f_1^2 - f_2 + 1\right) - \xi \ge 0\right\} \ge 0.8, \\
\xi \sim N(400, 0.5), x_1 \in [0, 1], x_2, \ldots, x_n \in [-2, 2], \\
N = 2, a = 1, n = 10.
\end{cases}
$$

D. CP4

$$
\min \ (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))
$$

$$
s.t., \begin{cases}
f_1(\boldsymbol{x}) = x_1 + \displaystyle\sum_{j\in J_1} h_j\left(y_j\right), f_2(\boldsymbol{x}) = 1 - x_1 + \sum_{j\in J_2} h_j\left(y_j\right), \\
J_1 = \{j \,|j \textit{ is odd and } 2 \le j \le n\}, \\
J_2 = \{j \,|j \textit{ is even and } 2 \le j \le n\}, \\
y_j = x_j - \sin\left(6\pi x_1 + \dfrac{j\pi}{n}\right), j = 2, \ldots, n, \\
h_2\left(t\right) = \begin{cases} |t|, & t < \frac{3}{2}\left(1 - \frac{\sqrt{2}}{2}\right), \\ 0.125 + (t-1)^2, & else, \end{cases} \\
h_j\left(t\right) = t^2, j = 3, 4, \ldots, n, n = 10.
\end{cases}
$$

E. CP5

$$
\min \ (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))
$$

$$
s.t., \begin{cases}
f_1(\boldsymbol{x}) = x_1 + \displaystyle\sum_{j\in J_1} h_j\left(y_j\right), f_2(\boldsymbol{x}) = 1 - x_1 + \sum_{j\in J_2} h_j\left(y_j\right), \\
J_1 = \{j \,|j \textit{ is odd and } 2 \le j \le n\}, \\
J_2 = \{j \,|j \textit{ is even and } 2 \le j \le n\}, \\
y_j = \begin{cases} x_j - 0.8x_1\cos\left(6\pi x_1 + \frac{j\pi}{n}\right), \textit{if } j \in J_1 \\ x_j - 0.8x_1\sin\left(6\pi x_1 + \frac{j\pi}{n}\right), \textit{if } j \in J_2 \end{cases}, \\
h_2\left(t\right) = \begin{cases} |t|, & t < \frac{3}{2}\left(1 - \frac{\sqrt{2}}{2}\right), \\ 0.125 + (t-1)^2, & else, \end{cases} \\
h_j\left(t\right) = 2t^2 - \cos\left(4\pi t\right) + 1 \ \ j = 3, 4, \ldots, n, \\
\Pr\left\{x_2 - 0.8x_1\sin\left(6\pi x_1 + \dfrac{2\pi}{n}\right) - 0.5x_1 + \zeta \ge 0\right\} \\
\quad \ge 0.8, \\
\xi \sim N(-1.1, 1), x_1 \in [0, 1], \\
\quad x_2, \ldots, x_n \in [-2, 2], n = 10.
\end{cases}
$$

F. CP6

$$
\min \quad (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))
$$

$$
s.t., \begin{cases} f_1(\boldsymbol{x}) = x_1 + \sum\limits_{j \in J_1} y_j^2, f_2(\boldsymbol{x}) = (1-x_1)^2 + \sum\limits_{j \in J_2} y_j^2, \\ \Pr\left\{x_2 - 0.8x_1 \sin\left(6\pi x_1 + \dfrac{2\pi}{n}\right) - \right. \\ sign\left(0.5\left(1-x_1\right) - \left(1-x_1\right)^2\right) \\ \left. \sqrt{\left|0.5\left(1-x_1\right) - \left(1-x_1\right)^2\right|} + \zeta_1 \geq 0\right\} \geq 0.8, \\ \Pr\left\{x_4 - 0.8x_1 \sin\left(6\pi x_1 + \dfrac{4\pi}{n}\right) - \right. \\ sign\left(0.25\sqrt{1-x_1} - 0.5\left(1-x_1\right)\right) \\ \quad \left. \sqrt{\left|0.25\left(1-x_1\right) - 0.5\left(1-x_1\right)\right|} + \zeta_2 \geq 0\right\} \geq 0.8, \\ \xi_1 \sim N(0,1), \xi_2 \sim N(0,1), x_1 \in [0,1], \\ \quad x_2, x_3, \ldots, x_n \in [-2,2], n = 10. \end{cases}
$$

G. CP7

$$
\min \quad (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))
$$

$$
s.t., \begin{cases} f_1(\boldsymbol{x}) = x_1 + \sum\limits_{j \in J_1} h_j\left(y_j\right), f_2(\boldsymbol{x}) = (1-x_1)^2 \\ + \sum\limits_{j \in J_2} h_j\left(y_j\right), \\ J_1 = \{j \,|\, j \textit{ is odd and } 2 \leq j \leq n\}, \\ J_2 = \{j \,|\, j \textit{ is even and } 2 \leq j \leq n\}, \\ y_j = \begin{cases} x_j - x_1 \cos\left(6\pi x_1 + \frac{j\pi}{n}\right), j \in J_1, \\ x_j - x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), j \in J_2, \end{cases} \\ h_2\left(t\right) = h_4\left(t\right) = t^2, h_j\left(t\right) = 2t^2 - \cos\left(4\pi t\right) + 1, \\ \quad j = 3, 5, 6, \ldots, \\ \Pr\left\{x_2 - \sin\left(6\pi x_1 + \dfrac{2\pi}{n}\right) - \right. \\ sign\left(0.5\left(1-x_1\right) - \left(1-x_1\right)^2\right) \\ \quad \left. \sqrt{\left|0.5\left(1-x_1\right) - \left(1-x_1\right)^2\right|} + \zeta_1 \geq 0\right\} \geq 0.8, \\ \Pr\left\{x_4 - \sin\left(6\pi x_1 + \dfrac{4\pi}{n}\right) - \right. \\ sign\left(0.25\sqrt{1-x_1} - 0.5\left(1-x_1\right)\right) \\ \quad \left. \sqrt{\left|0.25\left(1-x_1\right) - 0.5\left(1-x_1\right)\right|} + \zeta_2 \geq 0\right\} \geq 0.8, \\ \xi_1 \sim N(0,1), \xi_2 \sim N(0,1), x_1 \in [0,1], \\ \quad x_2, \ldots, x_n \in [-2,2], n = 10. \end{cases}
$$

ACKNOWLEDGMENT

Conflict of Interest: The authors declare that they have no conflict of interest.

REFERENCES

[1] A. Charnes and W. W. Cooper, "Chance-constrained programming," *Manage. Sci.*, vol. 6, pp. 73–79, Oct. 1959.

[2] S. Küçükyavuz and R. Jiang, "Chance-constrained optimization under limited distributional information: A review of reformulations based on sampling and distributional robustness," 2021, *arXiv:2101.08746*.

[3] R. Jiang, T. Sun, D. Liu, Z. Pan, and D. Wang, "Multi-objective reliability-based optimization of control arm using MCS and NSGA-II coupled with entropy weighted GRA," *Appl. Sci.*, vol. 11, no. 13, p. 5825, Jun. 2021.

[4] A. Geletu, A. Hoffmann, P. Schmidt, and P. Li, "Chance constrained optimization of elliptic PDE systems with a smoothing convex approximation," *ESAIM: Control, Optim. Calculus Variat.*, vol. 26, p. 70, 2020.

[5] K. Yang and Z. H. Zhang, "Adaptive sampling based immune optimization approach in noisy environments solving chance constrained programming," *J. Comput.*, vol. 25, no. 4, pp. 2–13, 2015.

[6] A. Prekopa, "On probabilistic constrained programming," in *Proc. Princeton Symp. Math. Program.* Princeton, NJ, USA: Princeton Univ. Press, 1970, pp. 113–138.

[7] X. Bai, J. Sun, and X. Zheng, "An augmented Lagrangian decomposition method for chance-constrained optimization problems," *INFORMS J. Comput.*, vol. 33, no. 3, pp. 1056–1069, Jul. 2021.

[8] B. Liu, *Theory and Practice of Uncertain Programming*. Berlin, Germany: Springer, 2009, pp. 349–363.

[9] A. Nemirovski and A. Shapiro, "Convex approximations of chance constrained programs," *SIAM J. Optim.*, vol. 17, no. 4, pp. 969–996, Jan. 2007.

[10] O. Klopfenstein and D. Nace, "A robust approach to the chance-constrained knapsack problem," *Operations Res. Lett.*, vol. 36, no. 5, pp. 628–632, Sep. 2008.

[11] A. Nemirovski and A. Shapiro, "Scenario approximations of chance constraints," in *Probabilistic and Randomized Methods for Design Under Uncertainty*. London, U.K.: Springer, 2006, pp. 3–47.

[12] J. Luedtke and S. Ahmed, "A sample approximation approach for optimization with probabilistic constraints," *SIAM J. Optim.*, vol. 19, no. 2, pp. 674–699, Jan. 2008.

[13] B. K. Pagnoncelli, S. Ahmed, and A. Shapiro, "Sample average approximation method for chance constrained programming: Theory and applications," *J. Optim. Theory Appl.*, vol. 142, no. 2, pp. 399–416, Aug. 2009.

[14] Z. Zhang, L. Wang, and F. Long, "Immune optimization approach solving multi-objective chance-constrained programming," *Evolving Syst.*, vol. 6, no. 1, pp. 41–53, Mar. 2015.

[15] S. Acharya, B. Belay, and R. Mishra, "Solving multi-objective chance constrained programming problem involving three parameters log normal distribution," *Int. J. Math. Oper. Res.*, vol. 18, no. 2, p. 236, 2021, doi: 10.1504/ijmor.2021.112939.

[16] K. Deb, S. Gupta, D. Daum, J. Branke, A. K. Mall, and D. Padmanabhan, "Reliability-based optimization using evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 1054–1074, Oct. 2009.

[17] E. Cantú-Paz, "Adaptive sampling for noisy problems," in *Proc. Genetic Evol. Comput. Conf.* Berlin, Germany. Springer, 2004, pp. 947–958.

[18] J. L. Higle and L. Zhao, "Adaptive and nonadaptive samples in solving stochastic linear programs: A computational investigation," *Stochastic Programming E-Print Series*, vol. 2005, p. 12, 2004.

[19] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on Stochastic Programming: Modeling and Theory*. Philadelphia, PA, USA: Society for Industrial Mathematics, 2009.

[20] C. A. Poojari and B. Varghese, "Genetic algorithm based technique for solving chance constrained problems," *Eur. J. Oper. Res.*, vol. 185, no. 3, pp. 1128–1154, Mar. 2008.

[21] S. He, S. S. Chaudhry, Z. Lei, and W. Baohua, "Stochastic vendor selection problem: Chance-constrained model and genetic algorithms," *Ann. Operations Res.*, vol. 168, no. 1, pp. 169–179, Apr. 2009.

[22] A. Udhayakumar, V. Charles, and M. Kumar, "Stochastic simulation based genetic algorithm for chance constrained data envelopment analysis problems," *Omega*, vol. 39, no. 4, pp. 387–397, Aug. 2011.

[23] J. Luedtke, "A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support," *Math. Program.*, vol. 146, nos. 1–2, pp. 219–244, Aug. 2014.

[24] V. Shabazbegian, H. Ameli, M. T. Ameli, G. Strbac, and M. Qadrdan, "Co-optimization of resilient gas and electricity networks; a novel possibilistic chance-constrained programming approach," *Appl. Energy*, vol. 284, Feb. 2021, Art. no. 116284.

[25] A. Yang, "Two-stage optimal scheduling model of microgrid based on chance-constrained programming in spot markets," *Processes*, vol. 8, no. 1, p. 107, 2020.

[26] P. K. Mianaei, M. Aliahmadi, S. Faghri, M. Ensaf, A. Ghasemi, and A. A. Abdoos, "Chance-constrained programming for optimal scheduling of combined cooling, heating, and power-based microgrid coupled with flexible technologies," *Sustain. Cities Soc.*, vol. 77, Feb. 2022, Art. no. 103502.

[27] C. Wang, W. Zhong, R. He, X. Peng, and L. Zhao, "A scenario-based chance-constrained program for gasoline blending under uncertainty," *Ind. Eng. Chem. Res.*, vol. 61, no. 15, pp. 5215–5226, Apr. 2022.

[28] S. Dong and M. Jiang, "A novel multi-objective lion swarm intelligent optimization based on cloud model for manufacturing scheduling problem," *J. Physics: Conf. Ser.*, vol. 2025, no. 1, Sep. 2021, Art. no. 012030.

[29] B. Odetayo, M. Kazemi, J. MacCormack, W. D. Rosehart, H. Zareipour, and A. R. Seifi, "A chance constrained programming approach to the integrated planning of electric power generation, natural gas network and storage," *IEEE Trans. Power Syst.*, vol. 33, no. 6, pp. 6883–6893, Nov. 2018.

[30] B. Liu, Q. Zhang, F. V. Fernández, and G. G. E. Gielen, "An efficient evolutionary algorithm for chance-constrained bi-objective stochastic optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 6, pp. 786–796, Dec. 2013.

[31] S. H. Nikokalam-Mozafar, B. Ashjari, R. Tavakkoli-Moghaddam, and A. Omidvar, "Solving a multi-objective chance-constrained hub covering location problem by discrete invasive weed optimization," *Cogent Eng.*, vol. 1, no. 1, Dec. 2014, Art. no. 991526.

[32] G. Tian, M. Zhou, and J. Chu, "A chance constrained programming approach to determine the optimal disassembly sequence," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 4, pp. 1004–1013, Oct. 2013.

[33] Y. Chen, Y. Li, B. Sun, C. Yang, and H. Zhu, "Multi-objective chance-constrained blending optimization of zinc smelter under stochastic uncertainty," *J. Ind. Manage. Optim.*, vol. 18, no. 6, p. 4491, 2022.

[34] A. Kamjoo, A. Maheri, A. M. Dizqah, and G. A. Putrus, "Multi-objective design under uncertainties of hybrid renewable energy system using NSGA-II and chance constrained programming," *Int. J. Electr. Power Energy Syst.*, vol. 74, pp. 187–194, Jan. 2016.

[35] Z. Zhang, L. Wang, and M. Liao, "Adaptive sampling immune algorithm solving joint chance-constrained programming," *J. Control Theory Appl.*, vol. 11, no. 2, pp. 237–246, May 2013.

[36] E. J. Hughes, "Constraint handling with uncertain and noisy multi-objective evolution," in *Proc. Congr. Evol. Comput.*, 2001, pp. 963–970.

[37] J. Y. Audibert, S. Bubeck, and R. Munos, "Bandit view on noisy optimization," in *Optimization for Machine Learning*, vol. 431. Cambridge, MA, USA: MIT Press, 2011.

[38] P. Matzinger, "Tolerance, danger and the extended family," *Annu. Rev. Immunology*, vol. 12, no. 1, pp. 991–1045, Jan. 1994.

[39] K. Yang, Z. Zhang, and J. Lu, "Adaptive racing ranking-based immune optimization approach solving multi-objective expected value programming," *Soft Comput.*, vol. 22, no. 7, pp. 2139–2158, Apr. 2018.

[40] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[41] Q. Zhang et al., "Multi-objective optimization test instances for the CEC 2009 special session and competition," Univ. Essex, Colchester, U.K., Tech. Rep. CES-487, 2008.

[42] M. Gong, L. Jiao, H. Du, and L. Bo, "Multiobjective immune algorithm with nondominated neighbor-based selection," *Evol. Comput.*, vol. 16, no. 2, pp. 225–255, Jun. 2008.

[43] L. Ma, N. Li, G. Yu, X. Geng, S. Cheng, X. Wang, M. Huang, and Y. Jin, "Pareto-wise ranking classifier for multiobjective evolutionary neural architecture search," *IEEE Trans. Evol. Comput.*, vol. 28, no. 3, pp. 570–581, Jun. 2024.

[44] N. Li, L. Ma, G. Yu, B. Xue, M. Zhang, and Y. Jin, "Survey on evolutionary deep learning: Principles, algorithms, applications, and open issues," *ACM Comput. Surveys*, vol. 56, no. 2, pp. 1–34, Feb. 2024.

KAI YANG received the B.S. degree from the Department of Mechanical Engineering, Guizhou University, Guizhou, China, in 1998, and the M.S. and Ph.D. degrees from the Department of Computer Science, Guizhou University, in 2008 and 2017, respectively. He is currently an Associate Professor with the Department of Information Engineering, Guizhou Communications Polytechnic, Guizhou. He has published more than ten academic papers. He has participated in three national level projects and presided over two provincial and ministerial level projects. His current research interests include immune optimization and stochastic programming.

RENCHONG ZHANG received the B.S. degree in mathematics and applied mathematics from Kaili University, Guizhou, China, in 2013, and the M.S. degree in computational mathematics from Guizhou University, Guizhou, in 2016. He is currently a Lecturer with the Computer and Information Engineering College, Guizhou University of Commerce, Guiyang, Guizhou. He has published more than eight academic papers. He has participated in one national level project, presided over one provincial and ministerial level project and one municipal level project, and participated in more than five other projects. His current research interests include intelligent optimization algorithm and stochastic programming.