Black-Box Optimization and Its Applications

6

Wataru Kumagai and Keiichiro Yasuda

1 Introduction

Super Smart Society (Society 5.0) (Cabinet Office 2021) and Bioeconomy (European Commission 2021) get attention recently as smarter or more advanced concept to realize pre-symptomatic state, healthy life expectancy, circular economy, and energy efficiency and conservation. To adapt to this trend, industry transformation is required, such as product design has more efficiency or manufacturing process is fundamental reformed by smart cells or genetics. Conversely, many systems become more large-scale and complex according to an appearance of *System of Systems* (SoS), which is a new kind of system consisting of connected various systems with operational autonomy and control autonomy (e.g., distribution system, medical system, power and energy system, or railway system).

From this background, the need for practical optimization is growing more urgent. Achieving this needs an integrated methodology by combination of not only on optimization theory but also surrounding technologies, such as simulation,

Adapted from Keiichiro Yasuda and Wataru Kumagai.
"Black-Box Optimization and Its Applications (written in Japanese)."
Journal of The Society of Instrument and Control Engineers, 59–12, 914/917 (2020).
Partly translated by permission of The Society of Instrument and Control Engineers.

W. Kumagai
Innovation Center, Marketing Headquarter, Yokogawa Electric Corporation, Tokyo, Japan
e-mail: Wataru.Kumagai@yokogawa.com

K. Yasuda (✉)
Graduate School of Systems Design, Tokyo Metropolitan University, Tokyo, Japan
e-mail: k-yasuda@tmu.ac.jp

T. Kaihara et al. (eds.), *Innovative Systems Approach for Facilitating Smarter World*, Design Science and Innovation,
https://doi.org/10.1007/978-981-19-7776-3_6

artificial intelligence-based (AI) modeling, and computing technologies; e.g., general-purpose graphics processing units (GPGPU). This type of optimization is data-driven approach using online data obtained from simulator or sensor, called black-box optimization (BBO). BBO uses only design-variable value and its objective-function value because the objective function is black-box or expensive function with its unknown landscape. Mathematical programming, which is a classical optimization algorithm's class using analytical information about the objective function (e.g., gradient or Hessian matrix) and its properties (e.g., convexity or variable dependency), cannot be applied to BBO. Therefore, BBO technology with an ability to adapt to quickly change of the environment surrounding optimization is required. This article focuses some representative approaches of BBO especially metaheuristics (Yang 2010). This article reviews its overview, desirable properties, constraint handling techniques, and its applications based on recent research trends.

Mathematical notations used in this article are as follows; $\mathbb{R}$ denotes the set of real numbers, N denotes the set of natural numbers, $\varnothing$ denotes the empty set, respectively. $[a,b]$, (a,b) denote the closed interval and the open interval between a and b, where $a, b \in \mathbb{R}$ $(a < b)$. The probability distributions $\mathcal{N}, \mathcal{U}(a,b)$ denote the multivariable standard normal (or gaussian) distribution and the continuous uniform distribution within $[a,b]$.

2 Black-Box Optimization

This article focuses on an unconstrained optimization problem minimizing an objective function $f(\boldsymbol{x})$, but Sect. 4 does on a constrained optimization problem minimizing the objective function $f(\boldsymbol{x})$ such that constraints $g_\kappa(\boldsymbol{x}) \leq 0$ $(\kappa = 1, 2, \ldots, K)$[1]; where $\boldsymbol{x} \in \mathbb{R}^N$ denotes the design variable, $f : \mathbb{R}^N \to \mathbb{R}$ denotes the objective function, and $g_\kappa(\boldsymbol{x}) : \mathbb{R}^N \to \mathbb{R}$ $(\kappa = 1, 2, \ldots, K)$ denotes the K constraint functions, respectively.

Direct-search or derivative-free method and sequential model-based optimization (SMBO) are global optimization or BBO approach. They search the global optima efficiently in black-box or expensive function using only design-variable value and its objective-function value, not the gradient to reduce a risk of getting trapped at low-grade local optima. While the direct-search or derivative-free method searches the optima in the objective function directly (e.g., metaheuristics or evolutionary algorithm), SMBO does it indirectly using a surrogate function sequentially, i.e., response surface methodology (Jones et al. 1998). The procedure of SMBO is the following:

1. Step 1: generating a surrogate function to approximate the objective function based on obtained data set of the design variables $\boldsymbol{x}$ and the objective-function values $f(\boldsymbol{x})$.

[1] Constraint condition is classified as unequally and equally constraints, but this article only focuses on unequally constraints because equally constraints are often relaxed to unequally constraints by introducing small tolerance.

2. Step 2: obtaining the temporary optima in the surrogate function.
3. Step 3: sampling based on the obtained optima.

For example, Bayes optimization or Kriging method (Snoek et al. 2012) is a well-known SMBO and iterates the following steps: generating a probabilistic surrogate function from sample data (e.g., Gaussian process regression); constructing an acquisition function corresponding to the surrogate model and searching its maxima (e.g., expected improvement); and sampling next candidate solutions around the maxima. SMBO is expected to advance by combination of AI-based modeling (e.g., the kernel method, ensemble learning, or deep learning); global optimization algorithm or the direct-search method[2]; and sampling techniques (Kawarabayashi and Yasuda 2008).

This article reviews the direct-search method as a core BBO technique, especially metaheuristics. Metaheuristics (or evolutionary algorithm) is a heuristic algorithm framework inspired by nature or physical phenomena; e.g., genetic algorithm and particle swarm optimization (Yang 2010). In metaheuristic algorithms, multiple search points (or population) search the global optima by updating or sampling iteratively in the solution space. $i = 1, 2, \ldots, m$ denotes the index of search points, $k \in N$ denotes the iteration counter, $m \in N$ denotes the number of search points, $\boldsymbol{x}^i(k) \in \mathbb{R}^N$ denotes the position of the i-th search point at k, respectively. They have the following features:

- *Approximation method*: which finds approximate solutions with highly optimality in a practical amount of time;
- *Stochastic method*: which uses pseudo-random numbers and can deal with uncertain values;
- *Multi-point type search method*: which has multiple search points interacting with each other.

Especially, they are expected as an effective approach to BBO from the viewpoint of the stochastic and multi-point type search method having advantages of reducing a risk of getting trapped at low-grade local optima in multimodal objective function, and benefiting from the evolution of parallel computing using GPGPU.

3 Robustness and Adaptability for Black-Box Optimization

This section provides (1) robustness and adaptability of metaheuristics as effective properties in BBO and (2) example algorithms with their properties.

[2] It needs global optimization for the surrogate function in SMBO to be complex or non-linear assuming that the objective function is non-convex or multimodal.

3.1 Robustness and Adaptability of Metaheuristics

It is important or required for users to select an appliable and appropriate algorithm and set its tunable parameters so that they are adapted to the class and analytical properties of the optimization problem. These properties make some difficulties as the following:

- *Separability (Decomposability) /Non-separability (Dependency):* whether there is an existence of cross term or interaction term between each coordinate of the search space in the objective function.
- *Well-Scaling (Well-Conditioning) /Ill-Scaling (Ill-Conditioning):* whether different directions in the search space show a largely different sensitivity in their contribution to the objective function.
- *Uni-Modal (Convex) /Multimodal (likely Non-Convex):* whether there are multiple local optimal solutions in the landscape of the objective function.
- *Origin-Independency/Origin-Dependency:* whether the search depends on translation of the origin in the solution space.

Conversely, BBO makes it impossible for users to know and use this information in advance. When applying metaheuristics to BBO, the search performance and parameter setting are affected by these difficulties. From this reason, users need to perform inefficient trial-and-error engineering. Therefore, it is desirable that metaheuristics for solving BBO have robustness and adaptability. In this context, the meaning of robustness and adaptability are the following inspired by modern control theory:

- *Robustness* is the algorithm's properties which the search performance is maintained as the problem's properties changing even with its parameter being fixed during the search process.
- *Adaptability* is the algorithm's ability of dynamically tuning its parameter so that it adapts to the problem's properties gradually during the search process.

Transformation invariance has been proposed as a property producing robustness. If we assume that a pseudo-random number sequence during the search process holds the same, transformation invariance is the quality of the search performance being reproducible with respect to transformation of the solution space or objective function (i.e., producing the same or similar results) (Hansen et al. 2011). Whether metaheuristic algorithm has transformation invariance depends on the case. When an algorithm lacks invariance to a certain transformation, it is shown that significant performance changes depending on this transformation. Transformation invariance is distinguished between two types: invariance to transformations of the solution space and the objective-function value f. Invariances giving an effectiveness of BBO are the following:

- *Similarity invariance*: which is invariant property to similarity transformation $T : \mathbb{R}^N \to \mathbb{R}^N$ of the search space; i.e., $T : \boldsymbol{x} \mapsto \alpha \boldsymbol{x}$, where $\alpha \in \mathbb{R}$.
- *Scale invariance:* which is invariant property to scaling $T : \mathbb{R}^N \to \mathbb{R}^N$ of the search space, including similarity transformation; i.e., $T : \boldsymbol{x} \mapsto \boldsymbol{D}\boldsymbol{x}$, where $\boldsymbol{D} \in \mathbb{R}^{N \times N}$ denotes a diagonal matrix.
- *Rotational invariance:* which is invariant property to rotation $T : \mathbb{R}^N \to \mathbb{R}^N$ of the search space; i.e., $T : \boldsymbol{x} \mapsto \boldsymbol{D}\boldsymbol{x}$, where $\boldsymbol{D} \in \mathbb{R}^{N \times N}$ denotes a rotation or orthonormal matrix.
- *Linear invariance:* which is invariant property to linear transformation $T : \mathbb{R}^N \to \mathbb{R}^N$ of the search space, including scaling and rotation; i.e., $T : \boldsymbol{x} \mapsto \boldsymbol{D}\boldsymbol{x}$, where $\boldsymbol{D} \in \mathbb{R}^{N \times N}$ denotes a representation matrix in linear algebra.
- *Translation invariance:* which is invariant property to translation $T : \mathbb{R}^N \to \mathbb{R}^N$ of the search space; i.e., $T : \boldsymbol{x} \mapsto \boldsymbol{x} - \boldsymbol{t}$, where $\boldsymbol{t} \in \mathbb{R}^N$ denotes a translation vector.
- *Affine invariance:* which is invariant property to affine transformation $T : \mathbb{R}^N \to \mathbb{R}^N$ of the search space, including linear transformation and translation; i.e., $T : \boldsymbol{x} \mapsto \boldsymbol{D}\boldsymbol{x} - \boldsymbol{t}$, where $\boldsymbol{D} \in \mathbb{R}^{N \times N}$, $\boldsymbol{t} \in \mathbb{R}^N$ denote a linear-transformation matrix and a translation vector, respectively.
- *Monotonic invariance of f*: which is invariant property to monotonic or order preserving transformation $T : \mathbb{R} \to \mathbb{R}$ of the objective-function value f; e.g., T satisfies $T(f_1) \leq T(f_2)$ whenever $f_1 \leq f_2$, where $f_1, f_2 \in \mathbb{R}$.

To deal with difficulties or necessary properties (e.g., ill-scalability, non-separability, and origin-dependency in the solution space), affine invariance (including scale invariance, rotational invariance, and translation invariance) is important because the search efficiency preserves regardless of them. Monotonic invariance of the objective function also is important because the search efficiency may preserve between non-convex (unimodal) and non-continuous objective function, and convex continuous objective function. Invariance is extremely useful in evaluating the performance of heuristic methods because it can generalize to the complete class of problems induced by the invariance.

Adaptive parameter tuning has been proposed as a function producing adaptability. *Intensification* and *diversification* are well-known search strategies in metaheuristics and lead to outstanding performance (Yang 2010). It is important to set and tune parameters appropriately because search dynamics can change depending on parameter. Parameter tuning is distinguished between the following two types inspired by system control theory (Kanemasa and Aiyoshi 2014):

- *Feedforward tuning:* which gives a parameter-tuning schedule before searching and tunes parameter following this.
- *Feedback tuning:* which gives a parameter-tuning rule in response to the search state or dynamics during the search process.

The search dynamics can be evaluated based on the search history of the set of search points. Therefore, adaptive parameter-tuning rule based on the search history is expected to reduce the effort in parameter setting and tuning and lead to excellent

performance. Kanemasa and Aiyoshi (2014) show the above concept of parameter tuning and design a new adaptive parameter rule providing excellent performance by genetic programming.

Additionally, parameter-tuning rules may have an influent on an algorithm's invariance. For this reason, invariance-based design guidelines for parameter-tuning rules are the following:

1. Guideline A: when an algorithm has a certain invariance, we add tuning rules so as to preserve the invariance;
2. Guideline B: when an algorithm lacks a certain invariance, we add tuning rules so as to compensate for the invariance.

Accordingly, metaheuristics with transformation invariance and adaptive parameter-tuning rules are thought to result in good robustness and adaptability for BBO.

3.2 Example Metaheuristics with Robustness and Adaptability

This subsection describes example metaheuristics following the above guidelines; the covariance matrix adaptation evolution strategy and the adaptive particle swarm optimization with rotational invariance.

3.2.1 Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

Evolution strategy (ES) is a metaheuristic algorithm inspired by the evolution process in biology or nature. In ES, a single search point moves through random walk distributed according to the normal distribution. The update rules of ES are expressed by Eqs. (6.1) and (6.2).

$$\hat{\boldsymbol{x}} = \boldsymbol{x}(k) + \sigma \boldsymbol{s} \tag{6.1}$$

$$\boldsymbol{x}(k+1) = \begin{cases} \hat{\boldsymbol{x}}; f(\hat{\boldsymbol{x}}) \leq f\left(\boldsymbol{x}(k)\right) \\ \boldsymbol{x}(k); \text{otherwise} \end{cases} \tag{6.2}$$

Here, $\boldsymbol{s} \in \mathbb{R}^N$ denotes a random vector distributed according to the multi-variable standard normal distribution $\mathcal{N}$, $\sigma > 0$ denotes the scale parameter. ES lacks similarity and scale invariance, so it is typically used with an adaptive parameter-tuning rule called *one-fifth success rule*, tuning the scale parameter σ according to the improvement frequency of the search point.

Conversely, Covariance Matrix Adaptation Evolution Strategy (CMA-ES) has been developed as a multi-point ES and various versions of CMA-ES have been proposed; e.g., $(\mu/\mu_w, \lambda)$-CMA-ES (Hansen et al. 2015). A search steps of a typical CMA-ES using weighted recombination, covariance matrix adaptation (CMA), and step-size adaptation (SSA) is the following:

1. Step 1: sampling the next position of search points from the normal distribution according to Eq. (6.3), which has a covariance matrix $\boldsymbol{C}(k) \in \mathbb{R}^{N \times N}$ and a scale parameter $\sigma(k) > 0$.

$$\boldsymbol{x}^i(k+1) = \boldsymbol{m} + \sigma(k)\boldsymbol{A}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{s}^i \tag{6.3}$$

2. Step 2: deriving a mean vector $\boldsymbol{m} \in \mathbb{R}^N$ using weighted recombination of the search points with superior objective-function value $f(\boldsymbol{x}^i(k+1))$.
3. Step 3: updating the parameters $\boldsymbol{C}(k)$ and $\sigma(k)$ according to the search state using CMA and SSA.

Here, $\boldsymbol{s}^i \in \mathbb{R}^N$ denotes a random vector distributed according to the multi-variable standard normal distribution $\mathcal{N}$, $\boldsymbol{A} = (\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_N) \in \mathbb{R}^{N \times N}$ denotes the basis transformation matrix, and $\boldsymbol{\Lambda} = \text{diag}\,(\lambda_1, \lambda_2, \ldots, \lambda_N) \in \mathbb{R}^{N \times N}$ denotes the eigen matrix, respectively. $\{(\boldsymbol{a}_1, \lambda_1), (\boldsymbol{a}_2, \lambda_2), \ldots, (\boldsymbol{a}_N, \lambda_N)\}(\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N)$ are pairs of the eigenvector and eigenvalue of the covariance matrix $\boldsymbol{C}(k)$.

CMA-ES has the following meaningful features for BBO:

- To have affine transformation invariance and monotonic invariance of the objective function f.
- To realize intensification and diversification-based search strategy by adaptive parameter-tuning rules, the search points move to a superior region obtained through the search process.
- To provide recommended values for all parameters.

Parameter-tuning rules of ES following the design guideline B in Subsection 3.1 give an equivalent effect of adding affine invariance to ES; e.g., the one-fifth success rule, CMA, and SSA. It is expected that CMA-ES has high search performance according to the fact that an advanced version of it is highly ranked in BBO competitions called *Black-Box Optimization Benchmarking workshop* (BBOB) (The Black-box Optimization Benchmarking (BBOB) n.d.).

3.2.2 Adaptive Particle Swarm Optimization with Rotational Invariance

Particle swarm optimization (PSO) (Yang 2010) is a metaheuristic algorithm inspired by swarm intelligence in biology. The update rules of PSO are expressed by the following equations:

$$\upsilon^i(k+1) = w\upsilon^i(k) + c_1 R_1\left(p^i(k) - \boldsymbol{x}^i(k)\right) + c_2 R_2\left(p^g(k) - \boldsymbol{x}^i(k)\right) \tag{6.4}$$

$$\boldsymbol{x}^i(k+1) = \boldsymbol{x}^i(k) + \upsilon^i(k+1) \tag{6.5}$$

$$\boldsymbol{p}^i(k) = \underset{\boldsymbol{x}^i(\kappa)}{\text{argmin}}\left\{ f\left(\boldsymbol{x}^i(\kappa)\right) | \kappa = 1, 2, \cdots, k \right\} \tag{6.6}$$

$$\boldsymbol{p}^g(k) = \underset{\boldsymbol{p}^i(k)\in\mathcal{P}(k)}{\operatorname{argmin}} f\left(\boldsymbol{p}^i(k)\right) \tag{6.7}$$

$$\mathcal{P}(k) = \left\{\boldsymbol{p}^i(k) | i = 1,2,\cdots,m\right\} \tag{6.8}$$

Here, w, c_1, $c_2 > 0$ denote PSO's parameters, $R = \text{diag}\,(R_{\ell 1}, R_{\ell 2}, \ldots, R_{\ell N}) \in \mathbb{R}^{N\times N}$ ($\ell = 1, 2$) denotes a random number matrix, and $R_{\ell n} (n = 1, 2, \ldots, N)$ denotes random number distributed according to the uniform distribution $\mathcal{U}(0,1)$, respectively. Especially, $\boldsymbol{p}^i(k) \in \mathbb{R}^N$ denotes a personal-best solution called *p-best*, $\boldsymbol{p}^g(k) \in \mathbb{R}^N$ denotes the group-best solution called *g-best*.

PSO has room for improvement in both of adaptability and robustness to various conditions. Yasuda et al. (2008) have proposed the swarm activity P as a metrics of the search state in terms of intensification and diversification, and derived accurately the stable and unstable regions in the PSO's parameter space using it. The swarm activity P is expressed by Eq. (6.9).

$$P(k) = \frac{1}{m\sqrt{N}} \sum_{i=1}^{m} \|\boldsymbol{v}^i(k)\| \tag{6.9}$$

Moreover, the activity feedback PSO (AFPSO) is proposed as a new-type PSO with an adaptive parameter-tuning rule to control the swarm activity P during the search process (Yasuda et al. 2008). The update rule of AFPSO consists of PSO and a tuning rule for the inertia parameter w. The tuning rules for $w(k)$ are expressed by Eqs. (6.10) and (6.11).

$$w(k+1) = w(k) + \text{sgn}\left(P(k) - P_t(k)\right)\Delta w \tag{6.10}$$

$$P_t(k) = \frac{\varepsilon_{\text{start}}}{\sqrt{N}} \left(\frac{\varepsilon_{\text{end}}}{\varepsilon_{\text{start}}}\right)^{\frac{k}{k_{\max}}} \|\boldsymbol{\gamma}_{\max} - \boldsymbol{\gamma}_{\min}\| \tag{6.11}$$

Here, $\boldsymbol{\gamma}_{\max}, \boldsymbol{\gamma}_{\min} \in \mathbb{R}^N$ denote the vectors determined from the initial search points, $k_{\max} \in \mathbb{N}$ denotes the iteration counter max, and $\text{sgn} : \mathbb{R} \to \{-1, 1\}$ denotes the sign function, respectively. After updating $w(k)$ with Eq. (6.10), we revise it so that $w(k) \in [w_{\min}, w_{\max}]$. This tuning rule adjusts $w(k)$ so that the swarm activity $P(k)$ follows a target value $P_t(k)$. It is shown that AFPSO has better search performance than PSO with various parameter-tuning rules (Yasuda et al. 2008).

However, Kumagai and Yasuda (2017, 2019) have pointed out that AFPSO has scale invariance, translation invariance, and monotonic invariance of f, but lacks rotational invariance such as PSO with various parameter-turning rules through mathematical proofs and numerical simulations. On that point, the adaptive PSO with rotational invariance using correlativity (adaptive CRI-PSO) has proposed as a new-type PSO with both of rotational invariance and the adaptive parameter-tuning rule (Kumagai and Yasuda 2019). The update rules of the adaptive CRI-PSO are basically the same as that of AFPSO, except that Eq. (6.4) is revised to Eq. (6.12).

$$\boldsymbol{v}^i(k+1) = w(k)\boldsymbol{v}^i(k) + c_1 \boldsymbol{A}\boldsymbol{R}_1\boldsymbol{A}^{\mathrm{T}}\left(\boldsymbol{p}^i(k) - \boldsymbol{x}^i(k)\right) + c_2\boldsymbol{A}\boldsymbol{R}_2\boldsymbol{A}^{\mathrm{T}}\left(\boldsymbol{p}^g(k) - \boldsymbol{x}^i(k)\right) \tag{6.12}$$

Here, $\boldsymbol{A} = (\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_N) \in \mathbb{R}^{N \times N}$ denotes the basis transformation matrix, and $\{\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_N\}$ are the eigenvectors of the covariance matrix based on the p-best distribution $\mathcal{P}(k)$. In the tuning rules for $w(k)$ of the adaptive CRI-PSO, parameters Δw, $\varepsilon_{\text{start}}$, ε_{end}, $w_{\min}$, $w_{\max} > 0$ are added.[3]

The adaptive CRI-PSO controls the swarm activity $P(k)$ while determining the perturbation direction from the p-best distribution $\mathcal{P}(k)$. In the above the parameter-tuning rule, controlling the search state according to the ideal ensures that the intensification and diversification-based search strategy is realized. This strategy is expected to give the adaptive CRI-PSO highly adaptation. Additionally, the adaptive CRI-PSO consists of Eq. (6.12) and the above parameter-tuning rule; and has similarity, rotational, and translation invariance in the solution space and monotonic invariance of f. But, because adding parameter-tuning rule preserves these invariances, this satisfies the design guideline A in Sect. 3.1. Thus, it is expected to have high robustness and adaptability. In fact, the search performance of the adaptive CRI-PSO is shown to be superior to that of PSO and AFPSO through numerical experiments (Yasuda et al. 2008).

CMA-ES and the adaptive CRI-PSO are common in the following points:

1. They get transformation invariance by using the covariance matrix based on the solution set obtained in the search process.
2. They get adaptability by parameter tuning according to the intensification and diversification-based search strategy.

4 Constrained Black-Box Optimization

The direct-search method including metaheuristics is considered to be unconstrained optimization problems, but we need to consider constraints in real-world use. This section describes classes of constraints in BBO and methods for handling them.

4.1 Classes of Constraints in BBO

This section focuses on constrained optimization problem in BBO. A solution satisfying all constraints is called *feasible solution*, and a solution not satisfying a constraint is called *infeasible solution* or *constraint-violating solution*. The notation $\mathcal{F}$ denotes the set of feasible solutions called *feasible region*.

[3] Kumagai and Yasuda (2019) given the following recommended values to for these parameters. $\gamma_{\max}$, $\gamma_{\min}$ are determined by the region for initial search points. $w_{\max}$, $w_{\min}$ should be to across the boundary line between stable and unstable regions in PSO's parameter space. It is pointed out that the boundary line is located inside the region $w \in [0.5,1.0]$ while $c_1 = c_2 = 1.4955$ through parameter analysis.

Constraints can be classified according to their analytic properties and formal properties. In terms of analytic properties, we can classify constraints as either linear constraints including non-negative constraints and upper/lower bound constraints, or non-linear constraints.[4] In terms of formal properties, we can classify constraints including simulation-based optimization or BBO based on the form. QRAK (Digabel and Wildy 2015) is a taxonomy rule with the four following perspectives:

- *Quantifiable (Q)/Nonquantifiable (N):* whether the degree of constraint violation can be quantified;
- *Relaxable (R)/Unrelaxable (U):* whether a constraint-violating solution can be evaluated;
- A priori *(A)/Simulation-based (S):* whether a simulation is needed to evaluate a solution;
- *Known (K)/Hidden (H):* whether the existence of constraints is known.

Constraints can be classified into 16 types according to the QRAK rule.[5]

This includes situations such as when constraint-violating solutions cannot be evaluated due to reasons on simulator's side, or when the simulation does not have completed successfully. In practical use, many cases are QRAK or QRSK constraints. The reason for this is that a binary constraint meaning whether a solution is feasible or infeasible can be quantified as numerical constraint violation, or constraint-violating solutions can be evaluated in some reasonable way. It would be meaningful to know classes of constraints in BBO more broadly and to consider handling methods matching them. Sections 4.2 and 4.3 will explain methods for handling QRAK or QRSK constraints.

4.2 Classical Method for Constraint Handling

Classical method for constraint handling is a type of problem-transformation method called the penalty-function method. If the class and properties of objective or constraint functions are clear, you can apply a method for constraint handling that matches the class of problem. For example, if the objective function and constraint functions are convex, a convex optimization method that uses analytic information about their functions (gradient or Hessian matrix) can solve this problem by constructing a new convex objective function introduced penalty function; and converting the problem to an unconstrained optimization, which called the augmented Lagrangian function. Equation (6.13) shows the augmented Lagrangian function $L : \mathbb{R}^N \to \mathbb{R}$ with a penalty function.

[4] Besides linear constraints and non-linear constraints, there are discrete constraints and level-sets constraints determined by objective-function values.

[5] Actually, the QRAK rule classifies constraints into nine types because it is not possible to calculate or quantify numerical constraint violation value if the existence of them is unknown.

$$L(x) = f(x) + \lambda\phi(x) \tag{6.13}$$

Here, $f : \mathbb{R}^N \to \mathbb{R}$ denotes the objective function, $\phi : \mathbb{R}^N \to \mathbb{R}$ denotes the penalty function, and $\lambda > 0$ denotes the penalty coefficient, respectively. Many definitions for penalty function ϕ are possible, but a typical definition is formulated as Eqs. (6.14) and (6.15).

$$\phi(x) = \sum_{k=1}^{k} \Omega_k(x) \tag{6.14}$$

$$\Omega_\kappa(\mathrm{x}) = \max\left\{g_\kappa(\mathrm{x}), 0\right\} \tag{6.15}$$

An amount of constraint violation $v(\boldsymbol{x})$, which amount of violating a constraint or all constraints, is basically defined by $v(\boldsymbol{x}) = \phi(\boldsymbol{x})$ and $v_\kappa(\boldsymbol{x}) = \Omega_\kappa(\boldsymbol{x})$.

4.3 Methods for Constraint Handling in BBO

4.3.1 Constraints Handling Techniques

The classical penalty-function method is not effective for constrained optimization problem including the black-box or non-convex objective function and constraint functions. Difficulties in this class of problem are the following:

1. Because there are multiple local optimal solutions in the feasible region $\mathcal{F}$ being non-convex or disconnected, it needs a global search capability.
2. Typically, because the global optimal solution is located on the boundary of the feasible region $\mathcal{F}$, it needs an efficient search on this boundary.

The constraint handling techniques (CHTs) have been developed as methods expanding the applicable scope of metaheuristics to this class of problems and cope with the above issues (Coello 2002; Mezura-Montes et al. 2011). Metaheuristic algorithm searches the global optimal solution by selection superior solutions. The selection operation evaluates and compares solutions obtained through multi-point search according to a metrics called fitness function $S(\boldsymbol{x})$. While the selection for unconstrained optimization uses the fitness S defined by the objective-function value f, CHTs use the fitness S defined by the objective-function value f and amount of constraint violation v. Additionally, it has been pointed out that using constraint-violating solutions is important for improving search efficiency in constrained BBO (Ray et al. 2009). According to the above discussion, CHTs can be largely classified depending on the following viewpoint:

1. CHTs can be categorized as penalty-based, separatist-based, or multi-objective-based method depending on the definitions of fitness S.
2. CHTs can be categorized as "none," "inexplicit," or "explicit" class depending on degree of utilization of constraint-violating solutions.

Table 6.1 Summary of studies on constraint handling techniques

–	Penalty-based CHT	Separatist-based CHT	Multi-objective-based CHT
None	Death penalty (Mezura-Montes and Coello 2011)	–	–
Inexplicit	Static penalty (Homaifar et al. 1994), Dynamic penalty (Joines and Houck 1994), Adaptive penalty (Lemonge and Barbosa 2004)	Feasibility rule (Mezura-Montes and Coello 2011), Stochastic ranking (Runarsson and Yao 2000), ε constraint method (Takahama and Sakai 2005), GCR (Runarsson and Yao 2003), HSR (Ho and Shimizu 2007), MCR (Garcia et al. 2017)	–
Explicit	ASCHEA (Hamida and Schoenauer 2000)	MCODE (Liu et al. 2007), TNSDM (Miyakawa et al. 2013)	Two-phase framework (Venkatraman and Yen 2005), IDEA (Ray et al. 2009), DeCODE (Wang et al. 2021), Adaptive Weighted MOEA/D (Yasuda et al. 2022)

Table 6.1 shows summary of studies on CHTs categorized based on these criteria. We will provide a description of each approach in next subsection and beyond.

4.3.2 Penalty-Based CHT

The penalty-based CHT uses the augmented Lagrangian function L defined in Eq. (6.13) as the fitness, but it differs from classical penalty functions in terms of the way it treats the penalty coefficient λ. The death penalty (Mezura-Montes and Coello 2011), static penalty (Homaifar et al. 1994), and dynamic penalty (Joines and Houck 1994) have been proposed as the penalty-based CHT. The death penalty assigns an extremely bad value (or an infinitely large value) to the penalty coefficient λ for constraint-violating solutions, and searches only using feasible solutions. The static penalty assigns a constant to λ and the dynamic penalty increases λ according to a given schedule during the search process. However, the search performance is deteriorated, and a feasible solution cannot be obtained without a properly set or tuning for the penalty coefficient λ. For example, the fitness function L highly depends on scale differences between constraint functions because of lacking monotonic invariance of the objective function and constraint functions if the amount of constraint violation v is defined in Eq. (6.14).

On that point, the adaptive penalty (Lemonge and Barbosa 2004) and ASCHEA (Hamida and Schoenauer 2000) have been proposed as the penalty-based CHT for adaptively tuning the penalty coefficient during the search process. In the adaptive penalties, a fitness function S of a constraint-violating solution $\boldsymbol{x}$ in a solution set $\mathcal{X}$ is formulated as Eqs. (6.16), (6.17), and (6.18).

$$S(x) = \hat{f}(x) + \sum_{\kappa=1}^{K} \lambda_\kappa \upsilon_\kappa(x) \tag{6.16}$$

$$\hat{f}(x) = \max\left\{f(x), \bar{f}\right\} \tag{6.17}$$

$$\lambda_\kappa = \left|\bar{f}\right| \frac{\bar{\upsilon}_k}{\sum_{\kappa=1}^{K} \bar{\upsilon}_k^2} \tag{6.18}$$

Here, the notations $\bar{f}$, $\bar{\upsilon}_\kappa$ denote the mean objective-function value f and the mean amount of constraint violation v_κ, respectively; and their mean values are averaged of f and v_κ of solutions in the solution set $\mathcal{X}$. Thus, it is possible to mitigate the above deficiency by assigning different penalty coefficients for each constraint function and adaptively tuning in response to the objective-function value and each amount of constraint violation obtained in the search process. However, because many penalty-based CHTs do not make use of the order relationship between the amount of constraint violation of solutions, it is thought that the usefulness of constraint-violating solutions be low.

4.3.3 Separatist-Based CHT

The separatist-based CHT separates objective-function values f and amount of constraint violation v, and evaluates and compares solutions; they can be classified as either switching-based or ranking-based CHT. The switching-based CHT selects and uses either of the objective-function value or the amount of constraint violation as a fitness S. Feasibility rule (Mezura-Montes and Coello 2011), stochastic ranking (Runarsson and Yao 2000), ε constraint method (ε CM) (Takahama and Sakai 2005), and MCODE (Liu et al. 2007) have proposed in this category. In ε CM, when comparing two solutions $\boldsymbol{x}$ and $\boldsymbol{y}$, the order relation of fitness S is defined in Eq. (6.19).

$$S(\boldsymbol{x}) \leq S(\boldsymbol{y}) \Leftrightarrow \begin{cases} f(\boldsymbol{x}) \leq f(\boldsymbol{y}); v(\boldsymbol{x}), v(\boldsymbol{y}) < \varepsilon \\ v(\boldsymbol{x}) \leq v(\boldsymbol{y}); \text{otherwise} \end{cases} \tag{6.19}$$

Here, $\varepsilon > 0$ denotes a threshold parameter for the amount of constraint violation and decreases to almost 0 gradually during the search process, since the search points improve the objective-function value after they move to near the feasible region early on.

The ranking-based CHT uses a fitness S based on a total rank in a solution set. The total rank consists of rank based on objective-function values and rank based on amount of constraint violation. Global competitive ranking (GCR) (Runarsson and Yao 2003), Ho-Shimizu ranking (HSR) (Ho and Shimizu 2007), multiple constraint ranking (MCR) (Garcia et al. 2017), and TNSDM (Miyakawa et al. 2013) have proposed in this category. In MCR, the fitness function $S(\boldsymbol{x})$ of a solution $\boldsymbol{x}$ in a solution set $\mathcal{X}$ is formulated as Eq. (6.20).

$$S(x) = \begin{cases} R_n + \sum_{\kappa=1}^{K} R_{v_\kappa} ; \mathcal{F} \cap \mathcal{X} = \varnothing \\ R_f + R_n + \sum_{\kappa=1}^{K} R_{v_\kappa} ; \text{otherwise} \end{cases} \tag{6.20}$$

Here, R_f, R_{v_κ}, R_n denote a rank of objective-function value f, ranks for each amount of constraint violation v_κ, and a rank of constraint violation count, respectively; and their ranks are determined from an order relation in the solution set $\mathcal{X}$. Because the separatist-based CHTs make use of the order relation between the amount of constraint violation, it is thought that they make better use of constraint-violating solutions than penalty-based CHT.

4.3.4 Multi-Objective-Based CHT

The multi-objective-based CHT converts a single-objective constrained optimization problem into a two-objective (objective-function value f and amount of constraint violation v) unconstrained optimization problem and uses evaluations and comparisons of the solution in multi-objective optimization. The terms of multi-objective optimization in the multi-objective-based CHT are explained below. Multi-objective optimization finds a Pareto optimum set in the objective-function space considering trade-off relationship between each objective function while the multi-objective-based CHT finds a feasible global optima in the space (f, v). *Superiority relation* (*Pareto dominance*) gives superiority or inferiority relation between two solutions in the space (f, v). When two solutions $\boldsymbol{x}, \boldsymbol{y}$ satisfy Eq. (6.21), an order relation between $\boldsymbol{x}$ and $\boldsymbol{y}$ is called "dominates $\boldsymbol{y}$" or "$\boldsymbol{x}$ is superior to $\boldsymbol{y}$."

$$\left(f(\boldsymbol{x}) \le f(\boldsymbol{y}) \wedge v(\boldsymbol{x}) \le v(\boldsymbol{y})\right) \wedge \left(f(\boldsymbol{x}) < f(\boldsymbol{y}) \vee v(\boldsymbol{x}) < v(\boldsymbol{y})\right) \tag{6.21}$$

In a solution set $\mathcal{X}$, a solution $\boldsymbol{x} \in \mathcal{X}$ to be not dominated by all other solutions $\boldsymbol{y} \in \mathcal{X}$ is called *non-dominated solution* or *Pareto solution*. A set of non-dominated solutions in the solution space is called *Pareto optimum set* or *Pareto frontier*. A feasible solution is located on the f-axis in the space (f, v) and means *weakly Pareto solution*. Note that the multi-objective-based CHT aims to find the only feasible global optima located on the intersection of the Pareto frontier and the f-axis in the space (f, v), not the entire of this region.

The multi-objective-based CHT can be classified as either the Pareto ranking-based or decomposition-based CHT in the viewpoint of multi-objective optimization approaches. *Pareto ranking* gives candidate solutions a rank-based fitness constructed by superiority relation and crowded distance in the objective-function space and is used in NSGA-II (Deb et al. 2002) for multi-objective optimization. The Pareto ranking-based CHT uses the fitness based on Pareto ranking in the space (f, v). However, even if feasible solutions are obtained in the search process, they are easily eliminated because they are weakly Pareto solutions. On that point, as the Pareto ranking-based CHT preserving some feasible solutions during the search process, the two-phase framework (Venkatraman and Yen 2005) and the

infeasibility driven evolutionary algorithm (IDEA) (Ray et al. 2009) have been proposed. IDEA preserves both of feasible solutions and constraint-violating solutions in the search points and assigns two-types fitness for each solution set. One fitness of feasible solutions is the objective-function value f, and the other fitness of constraint-violating solutions is a rank R_p based on Pareto ranking. In IDEA, when comparing two feasible solutions or two constraint-violating solutions, the order relation of fitness S is defined in Eq. (6.22).

$$S(\boldsymbol{x}) \leq S(\boldsymbol{y}) \Leftrightarrow \begin{cases} f(\boldsymbol{x}) \leq f(\boldsymbol{y}); \boldsymbol{x}, \boldsymbol{y} \in \mathcal{F} \\ R_p(\boldsymbol{x}) \leq R_p(\boldsymbol{y}); \boldsymbol{x}, \boldsymbol{y} \notin \mathcal{F} \end{cases} \tag{6.22}$$

Decomposition decomposes a multi-objective optimization problem into a set of single-objective optimization subproblems and solves them in parallel by a scalar-based fitness function with a different weight parameter $w^i \in [0, 1]$ assigned for each search point $\boldsymbol{x}^i$; and is used in MOEA/D (Zhang and Li 2007) for multi-objective optimization. The decomposition-based CHT uses the above problem decomposition in the space (f, v). The scalarizing function S of the weighted sum is formulated as Eq. (6.23).

$$S(\boldsymbol{x}; w) = wf(\boldsymbol{x}) + (1-w)v(\boldsymbol{x}) \tag{6.23}$$

Note that MOEA/D for multi-objective optimization assigns the weights to constant value during the search process and distributes them uniformly in the objective-function space to find the entire of the Pareto frontier, but this is not effective of constrained optimization to find the only feasible global optima finally. On that point, as the decomposition-based CHT with weight-tuning rule dynamically during the search process, DeCODE (Wang et al. 2021), and the adaptive weighted MOEA/D (Yasuda et al. 2022) have been proposed. DeCODE uses a weight-tuning rule dynamically so that the search region is limited from the violation region to the feasible region gradually. The adaptive weighted MOEA/D uses a weight-tuning rule adaptively so that the search region is balanced to cover the boundary of the feasible region $\mathcal{F}$, and the tuning rules for w^i are expressed by Eqs. (6.24) and (6.25).

$$w^i(k) = \alpha(k)\frac{i-1}{m-1} \tag{6.24}$$

$$\alpha(k+1) = \begin{cases} \gamma_{\mathrm{u}}\alpha(k); \boldsymbol{x}^t(k) \in \mathcal{F} \\ \gamma_{\mathrm{d}}\alpha(k); \text{otherwise} \end{cases} \tag{6.25}$$

Here, $\alpha \in [0, 1]$ denotes a variable parameter for controlling the distribution of weights w^i in the space (f, v), $\boldsymbol{x}^t(k)$ denotes the search point having t-th weight w^t. $\gamma_{\mathrm{u}} > 1$, $\gamma_{\mathrm{d}} \in (0, 1)$ denote an increasing coefficient and a decreasing coefficient, respectively. After updating w^i with Eq. (6.24), we revise it so that $w^i(k) \in [\delta, 1]$, and after updating α by Eq. (6.25), we revise it so that $\alpha(k) \in [0, 1]$.[6]

[6]Yasuda et al. (2022) given the following recommended values to for these parameters; $\alpha(1) = 1.0$, $t = \lfloor 0.8m \rfloor$, $\gamma_{\mathrm{u}} = 1.001$, $\gamma_{\mathrm{d}} = 0.999$, $\delta = 10^{-15}$.

The multi-objective-based CHT makes the most utilization of constraint-violating solutions than the penalty-based and the separatist-based CHTs because an effective search works on trade-off regions between objective-function value f and amount of constraint violation v by superiority relation. The Pareto ranking-based and the decomposition-based CHTs have the following advantages and disadvantages:

- Pareto ranking-based CHT:
  - Advantage: it is possible to search the trade-off region while not being affected by a difference between f and v scales; or a shape of the Pareto frontier because the fitness S is determined only from ranks in the space (f, v).
  - Disadvantage: it is difficult to gain feasible solutions if the Pareto frontier is large in the solution space, which the feasible region is narrower than the violation region, because it searches the entire of the region uniformly.
- Decomposition-based CHT:
  - Advantage: it is possible to improve an ability to gain feasible solutions by weight tuning so that the search region is limited to a part of the Pareto frontier because solutions converge almost surely and the weights corresponds to the Pareto solutions basically.
  - Disadvantage: it is difficult to maintain the limitation in the space (f, v) as ideal and the corresponding relation between the weights and the Pareto solutions if the difference between f and v scales is large (Ishibuchi et al. 2017); or a scalarization function S does not match the shape of the Pareto frontier near the optimal solution.[7]

Both approaches are complementary, but there is a difference in effectiveness on constrained optimization. The Pareto ranking-based CHT preserves some feasible solutions in the search process, but it is relatively disadvantaged in finding the only feasible global optima because the search region cannot be limited to the part of the Pareto frontier. In contrast, the decomposition-based CHT can search the limited region, and its disadvantage is considered less serious than the Pareto ranking-based CHT according to the following reasons:

1. The difference between f and v scales can be suppressed by normalization or standardization (Ishibuchi et al. 2017).
2. Even if the uniformity of approximation to the Pareto frontier is reduced a little, there is some effectiveness.

In fact, Yasuda et al. (2022) confirmed that the adaptive weighted MOEA/D is superior to the separatist-based CHT and the Pareto ranking-based CHT in terms of both of convergence on a feasible solution and global optimization performance through numerical simulation. From the above, the decomposition-based CHT is

[7] The scalarizing function of the weighted sum cannot approximate the Pareto frontier uniformly, whose shape is not convex (non-convex or disconnected) (Zhang and Li 2007).

considered advantageous in constrained optimization in terms of the ability to gain feasible solutions.

4.3.5 General Comment

According to the above review, because CHTs use the order relation in the solution set obtained in the search process, they have a high affinity for metaheuristics with multi-point and direct-search method. As shown in Table 6.1, CHTs can be largely classified based on the viewpoint of definition of fitness function and degree of utilization of constraint-violating solutions; and especially the multi-objective-based CHT makes the most utilization of them. But the fitness function S in CHTs is deeply related to intensification and diversification-based search strategy, and transformation invariance; e.g., the monotonic invariance of objective function f and each constraint function g_k. In the future research, it is desirable to develop and combine metaheuristic algorithms and CHTs for constrained BBO considering the above strategy or property totally.

5 Application of BBO

This section reviews three important examples of BBO applications. First example is online operation optimization or configuration design for SoS. SoS consists of connected various systems with operational autonomy and control autonomy (e.g., distribution system, medical system, power and energy system, or railway system). This needs total and data-driven optimization using obtained online and uncertain data from sensor and state of each system in real time. Demand response maintains the supply-demand balance efficiently by providing people with data and financial incentives to indirectly change their behavior and demand, such as dynamic pricing in energy management systems (Yasuda and Tokoro 2020). Even if a state of SoS involves uncertainly such as potential to change people's behavior or demand, BBO can be applied to it. And there is an approach to find and design configuration system of SoS (Agarwal et al. 2015). These problems are formulated as constrained BBO.

Second example is hyperparameter tuning or optimization for expensive machine learning models. Automated machine learning (AutoML) frameworks and deep learning (deep neural networks) have many tunable hyperparameters such as very complex parameter architecture (Feurer and Hutter 2019). On that point, parameter-tuning techniques have proposed for automating their engineering process. It is shown that these techniques are superior to manually tuning or engineering by human in the viewpoint of performance and time comsuming (Snoek J et al. 2012; Bergstra et al. 2011); especially neural architecture search is the most successful example (Elsken et al. 2019). Hyperparameter tuning problem is formulated as constrained BBO.

Third example is material discovery, design, and optimization for multi-range application. Design of new and highly functional material including raw material or chemical compounds can change all manufacturing areas, such as healthcare, medical care, and energy. Industrial biotechnology can realize the production of

renewable biological resources and the conversion of these resources and waste streams into value added products, which alternative energy sources, biopharmaceutical, and high-quality food, by smart genetics and cells. The bio-industry subcommittee of Ministry of Economy, Trade, and Industry (METI) in Japan shows the following examples as their use cases (Ministry of Economy, Trade, and Industry (METI) n.d.):

1. Regenerative medicine or gene therapy may cause radical treatments for diseases.
2. In manufacturing process, redesign of cell's function may improve efficiency of industrial production; e.g., transformation from glucose to raw material for high-quality plastics.

Material discovery problem for these processes is formulated as constrained BBO (Terayama et al. 2021).

6 Conclusion

This article reviewed the main approaches to BBO, especially metaheuristics and constraint handling techniques, and provided some applications of BBO. As we shift toward Super Smart Society (Society 5.0) or Bioeconomy, real-world systems will grow in scale and complexity such as SOS. Conversely, practical system optimization will realize to combine surrounding technologies advancing rapidly. It is desirable to develop BBO with an ability to adapt all of these changes. Given these trends, we believe that SMBO for constrained BBO will be increasingly important, and these technologies should be able to adapt to various applications such as SOS in the future.

References

Agarwal S, Pape LE, Dagli CH, Ergin NK, Enke D, Gosavi A, Qin R, Konur D, Wang R, Gottapu RD (2015) Flexible and intelligent learning architectures for SoS (FILA-SoS): architectural evolution in systems-of-systems. Procedia Computer Sci 44:76–85

Bergstra J, Bardenet R, Bengio Y, Kégl B (2011) Algorithms for hyper-parameter optimization. Adv Neural Inf Proces Syst 24:2546–2554

Cabinet Office, Government of Japan (2021) Society 5.0. https://www8.cao.go.jp/cstp/english/society5_0/index.html. Accessed 15 May 2022

Coello CAC (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191(11–12):1245–1287

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

Digabel SL, Wildy SM (2015) A taxonomy of constraints in simulation-based optimization. Les cahiers du GERAD, technical report G-2015-57. https://doi.org/10.48550/arXiv.1505.07881

Elsken T, Metzen JH, Hutter F (2019) Neural architecture search, Automated machine learning. In: Hutter F, Kottohoff L, Vanschoren J (eds) The Springer series on challenges in machine learning, Cham, Springer, pp 63–77 https://doi.org/10.1007/978-3-030-05318-5_3

European Commission (2021) What is the Bioeconomy. https://ec.europa.eu/research/bioeconomy/policy/bioeconomy_en.htm. Accessed 15 May 2022

Feurer M, Hutter F (2019) Hyperparameter optimization. In Hutter F, Kotthoff L, Vanschoren J (eds): Automated machine learning. In: The Springer series on challenges in machine learning. Cham, Springer, pp. 3–33 https://doi.org/10.1007/978-3-030-05318-5_1

Garcia R, Lima B, Lemonge A, Jacob B (2017) A rank-based constraint handling technique for engineering design optimization problems solved by genetic algorithms. Comput Struct 187:77–87

Hamida SB, Schoenauer M (2000) An adaptive algorithm for constrained optimization problems. In: Proceedings of the International Conference on Parallel Problem Solving from Nature. pp. 529–538

Hansen N, Ros R, Mauny N, Schoenauer M, Auger A (2011) Impacts of invariance in search: when CMA-ES and PSO face ill-conditioned and non-separable problems. Appl Soft Comput 11(8):5755–5769

Hansen N, Arnold DV, Auger A (2015) Evolution strategies. In: Kacprzyk J, Pedrycz W (eds) Springer handbook of computational intelligence, Springer, vol 44, Berlin, Heidelberg, pp 871–898 https://doi.org/10.1007/978-3-662-43505-2_44

Ho PY, Shimizu K (2007) Evolutionary constrained optimization using an addition of ranking method and a percentage-based tolerance value adjustment scheme. Inf Sci 177(14):2985–3004

Homaifar A, Qi CX, Lai SH (1994) Constrained optimization via genetic algorithms. Simulation 62(4):242–253

Ishibuchi H, Doi K, Nojima Y (2017) On the effect of normalization in MOEA/D for multi-objective and many-objective optimization. Complex Intell Sys 3(4):279–294

Joines J, Houck C (1994) On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GA's. In: Proceedings of the First IEEE Conference on Evolutionary Computation. pp. 579–584 https://doi.org/10.1109/ICEC.1994.349995

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. J Glob Optim 13:455–492

Kanemasa M, Aiyoshi E (2014) Algorithm tuners for PSO methods and genetic programming techniques for learning tuning rules. IEEJ Trans Electr Electron Eng 9(4):407–414

Kawarabayashi M, Yasuda K (2008) Integrated optimization method using particle swarm optimization and Modeling (written in Japanese). Trans Soc Instrum Control Eng 44(11):855–862

Kumagai W, Yasuda K (2017) Making rotational invariance of particle swarm optimization based on correlativity. IEEJ Trans Electr Electron Eng 12(S2):S131–S132

Kumagai W, Yasuda K (2019) Adaptive particle swarm optimization with rotational invariance (written in Japanese). Trans Inst Electr Eng Japan Part C 139(10):1201–1214

Lemonge A, Barbosa H (2004) An adaptive penalty scheme for genetic algorithms in structural optimization. Int J Numer Methods Eng 59(5):703–736

Liu B, Ma H, Zhang X, Zhou Y (2007) A memetic Coevolutionary differential evolution algorithm for constrained optimization. In: Proceedings of the 2007 IEEE congress on evolutionary computation. pp. 2996–3002 https://doi.org/10.1109/CEC.2007.4424853

Mezura-Montes E, Coello CAC (2011) Constraint-handling in nature-inspired numerical optimization: past, present and future. Swarm Evol Comput 1(4):173–194

Ministry of Economy, Trade, and Industry (METI) (n.d.), Bio-Industry Subcommittee's Report (2021) Fifth industrial revolution. Cultivated with Biotechnology. https://www.meti.go.jp/english/press/2021/0202_001.html. Accessed 2 May 2022

Miyakawa M, Takadama K, Sato H (2013) Two-stage nondominated sorting and directed mating for solving problems with multi-objectives and constraints. In: proceedings of the 2013 genetic and evolutionary computation conference. pp. 647–654 https://doi.org/10.1145/2463372.2463449

Ray T, Singh HK, Isaacs AA, Smith W (2009) Infeasibility driven evolutionary algorithm for constrained optimization. In: Mezura-Montes, E. (eds) Constraint-handling in evolutionary opti-

mization. Studies in Computational Intelligence, vol 198. Springer, Berlin, Heidelberg. pp. 145–165 https://doi.org/10.1007/978-3-642-00619-7_7
Runarsson TP, Yao X (2000) Stochastic ranking for constrained evolutionary optimization. IEEE Trans Evol Comput 4(3):284–294
Runarsson TP, Yao X (2003) Constrained evolutionary optimization-the penalty function approach. In: Sarker R, Mohammadian M, Yao X (eds) Evolutionary optimization. International series in Operations Research & Management Science, vol 48. Springer, Boston, MA, pp 87–113
Snoek J, Larochelle H, Adams RP (2012) Practical Bayesian optimization of machine learning algorithms. Adv Neural Inf Proces Syst 25(2):2951–2959
Takahama T, Sakai S (2005) Constrained optimization by ε constrained particle swarm optimizer with ε-level control. In: Proceedings of the 4th IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology. pp. 1019–1029
Terayama K, Sumita M, Tamura R, Tsuda K (2021) Black-box optimization for automated discovery. Acc Chem Res 54(6):1334–1346. https://doi.org/10.1021/acs.accounts.0c00713
The Black-box Optimization Benchmarking (BBOB) (n.d.) Workshop. http://numbbo.github.io/workshops/index.html. Accessed 29 March 2022
Venkatraman S, Yen GG (2005) A generic framework for constrained optimization using genetic algorithms. IEEE Trans Evol Comput 9(4):424–435
Wang B, Li H, Zhang Q, Wang Y (2021) Decomposition-based multiobjective optimization for constrained evolutionary optimization. IEEE Transact Sys Man Cybernet: Syst 51(1):574–587
Yang XS (2010) Nature-inspired metaheuristic algorithms, 2nd edn. Luniver Press
Yasuda K, Tokoro K (2020) Smartification of social infrastructure for efficient power and energy use. In: Kaihara T, Kita H, Takahashi S (eds) Innovative systems approach for designing smarter world. Springer, Singapore, pp 133–145
Yasuda K, Iwasaki N, Ueno G, Aiyoshi E (2008) Particle swarm optimization: a numerical stability analysis and parameter adjustment based on swarm activity. IEEJ Trans Electr Electron Eng 3(6):642–659
Yasuda Y, Kumagai W, Tamura K, Yasuda K (2022) An extension of MOEA/D to constrained optimization and adaptive weight adjustment (written in Japanese). Trans Inst Electr Eng Japan Part C 142(1):108–109
Zhang Q, Li H (2007) MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 11(6):712–731

Wataru Kumagai Born in 1991, Wataru Kumagai completed his doctoral program in Electrical and Electronic Engineering at the Graduate School of Science and Engineering, Tokyo Metropolitan University, Japan, in 2020. Since 2016, he has been working at Yokogawa Electric Corporation, Japan. He is engaged in research on systems optimization (evolutionary computation), machine learning, and energy optimization. He holds a Doctor of Engineering and is a member of The Institute of Electrical Engineers of Japan.

Keiichiro Yasuda Born in 1960, Keiichiro Yasuda completed his doctoral program in Electrical Engineering at the Graduate School of Engineering, Hokkaido University, in 1989. The same year, he became an assistant professor in the Faculty of Engineering at the Tokyo Metropolitan University. In 1991, he became an associate professor in the Faculty of Engineering at the Tokyo Metropolitan University and, since 2006, he has been a professor in the Graduate School of Science and Engineering. He is engaged in research on systems optimization and power systems engineering. He holds a Doctor of Engineering and is a member of societies such as The Society of Instrument and Control Engineers, The Institute of Electrical Engineers of Japan, The Japanese Society for Evolutionary Computation, and the IEEE.