Advanced Golden Jackal Optimization for Solving the Constrained Integer Stochastic Optimization Problems

Shih-Cheng Horng schong@cyut.edu.tw

Shieh-Shing Lin sslin@mail.sju.edu.tw

Revised and Resubmitted to the **Mathematics and Computers in Simulation**

as a **REGULAR PAPER**

Correspondent: Professor Shih-Cheng Horng

Institute: Department of Computer Science & Information Engineering

Chaoyang University of Technology

e-mail : schong@cyut.edu.tw

Shih-Cheng Horng is a professor of the Department of Computer Science & Information Engineering at Chaoyang University of Technology, Taiwan, R.O.C. Shieh-Shing Lin is a professor of the Department of Electrical Engineering at St. John's University, Taiwan, R.O.C. This research is supported in part by the National Science and Technology Council in Taiwan, R.O.C., under Grant NSTC111-2221-E-324-021.

Abstract

Constrained integer stochastic optimization problems (CISOP) are optimization problems where the cost function is stochastic and the constraints are deterministic. Solving the CISOP using conventional optimization techniques is time-consuming when the design space is huge. Ordinal optimization offers an efficient technique suitable for solving the CISOP. In this paper, an approach that uses golden jackal optimization assisted by ordinal optimization (GJOO) is developed for resolving the CISOP in a relatively short time. The GJOO is composed of three phases: metamodel, global search, and ranking and selection. At first, the polynomial chaos expansion is used as a metamodel to rapidly assess a solution. Next, advanced golden jackal optimization is adopted to determine N excellent solutions from the entire design space. At last, the modified optimal computing budget allocation is adopted to determine a distinguished solution from the N excellent solutions. The GJOO is utilized to the shipping/receiving docks optimization problem of a container freight station in air cargo. The performances of the GJOO are compared with those of five meta-heuristic methods. Empirical results demonstrate the efficiency and robustness of the GJOO algorithm.

Keywords: polynomial chaos expansion, golden jackal optimization, ordinal optimization, optimal computing budget allocation, container freight station, shipping/receiving docks.

1. Introduction

Constrained integer stochastic optimization problems (CISOP) are optimization problems where the cost function is stochastic and the constraints are deterministic [1]. CISOP appear in almost all fields, such as periodic inspection inventory systems, buffer capacity allocation, pull-product systems, stochastic economic lot scheduling, and shipping/receiving docks optimization of a container freight station. The CISOP belong to NP-hard classes, which may not resolve comprehensive solutions within a reasonable time [2]-[3].

Three classical methods are commonly used to resolve CISOP: stochastic approximation [4], sample-path method, and sample average approximation [5]. Nonetheless, trapping in the local optimum and slow convergence rates are two main disadvantages of the above approaches. Heuristic algorithms are other methods adopted to resolve CISOP, such as genetic algorithm (GA) [6], tabu search (TS) [7], and simulated annealing (SA) [8]. However, heuristic methods will probably not yield the optimal decision because of miscalculations. Quick decisions without all the information will lead to errors in judgment. Nowadays, swarm-based algorithms have been extensively applied to solve CISOP [9]. Swarm-based algorithms is a family of nature inspired algorithms that can produce low cost and robust solutions to various optimization problems. They are inspired by the social behavior of gregarious insects and other animals. Food provision has been the source of inspiration for the development of novel algorithms, such as golden jackal optimization (GJO) [10]-[11], Harris hawks optimization [12]-[13], white shark optimizer [14], African vultures optimization algorithm [15], sine-cosine and spotted hyena-based chimp optimization algorithm [16], orca predation algorithm [17], snake optimizer [18], remora optimization algorithm [19], coati optimization algorithm [20], artificial hummingbird algorithm [21], red fox optimization [22], tunicate swarm algorithm [23], reptile search algorithm [24], serval optimization algorithm [25], multi-cohort whale optimization [26], and termite life cycle optimizer [27].

Swarm-based algorithms have been proven to be a comprehensive approach to solve complex optimization problems.

The CISOP are extremely difficult due to three issues: (i) a large design space, (ii) constraints must be simultaneously satisfied, and (iii) exactly evaluating a cost function is time-consuming. The ordinal optimization (OO) theory [28] is presented to address issues (i) to (iii) at the same time. The OO theory is composed of two basic ideas: sorting comparison and goal softening. Sorting comparison only uses comparisons between items to gain information about the relative order of items. Goal softening only needs to obtain good enough solutions, rather than the solution that is the best for sure. The OO theory has been adopted quite successfully to some complex optimization problems, such as routing optimization in queueing networks [29], optimal shortcuts of sorting conveyor system [30], job-shop scheduling [31], and optimal staff in emergency department healthcare [32].

Although the OO theory can narrow down the design space and speed up the searching procedure, the stochastic cost function still apparently influences computational performance. To resolve the CISOP in a short time, an algorithm that uses golden jackal optimization assisted by ordinal optimization (GJOO) is developed to look for a distinguished solution. The OO theory uses rough evaluation to derive an excellent subset for which simulations are necessary and worthwhile to find distinguished solutions with a substantially reduced computational burden. The GJOO comprises three parts: the metamodel, global search, and ranking and selection. At first, the polynomial chaos expansion (PCE) [33] is used as a metamodel to rapidly estimate a solution. Next, advanced golden jackal optimization (AGJO) is developed to look for N excellent solutions from the entire design space. At last, the modified optimal computing budget allocation (MOCBA) is adopted to seek a distinguished solution from the N excellent solutions. The three parts apparently diminish the computing time needed to solve the CISOP.

Afterward, the GJOO is applied to the shipping/receiving docks optimization problem of a container freight station in air cargo, which can be formulated as a CISOP. The goal of this CISOP is to look for the optimal shipping/receiving docks of a container freight station for minimizing the long-run average waiting time of a truck. The main contributions of the paper are summarized as follows:

(1) The GJOO algorithm is developed to find a distinguished solution to CISOP in a reasonable time.

(2) The GJOO algorithm is applied to seek the optimal shipping/receiving docks for the container freight station.

(3) The OO theory can be adopted to assist novel optimization approaches to resolve more complex stochastic optimization problems.

The structure of this paper is as follow. Section 2 presents the CISOP and illustrates the GJOO algorithm to seek a distinguished solution. Section 3 introduces the shipping/receiving docks optimization problem of a container freight station in air cargo, that is formulated as a CISOP. The GJOO algorithm is applied to solve this CISOP. Section 4 portrays experiential results performed to investigate the performance of the proposed approach. Section 5 concludes the paper with future scope.

2. Merging Golden Jackal Optimization with Ordinal Optimization

2.1 Constrained Integer Stochastic Optimization Problems

There are two challenges to CISOP. (i) The feasibility of a solution cannot be known with certainty due to the random nature of the cost function. (ii) The design space has no appropriate structural information to identify the optimal solution. The CISOP can be formulated mathematically, as shown below.

$$\min \ \mathrm{E}\left[f(\mathbf{x}) \right] \tag{1}$$

$$\text{subject to} \ \ g_i(\mathbf{x}) = c_i, i = 1, \ldots, I \ , \tag{2}$$

$$h_j(\mathbf{x}) > d_j,\ j = 1,\ldots,J\ . \tag{3}$$

$$\mathbf{V} \le \mathbf{x} \le \mathbf{U}\ . \tag{4}$$

where $\mathbf{x} = [x_1,\ldots,x_K]^T$ is a solution, $\mathrm{E}\left[f(\mathbf{x})\right]$ is the expectation of the cost function, $g_i(\mathbf{x})$ is the *i*th equality constraint, c_i is specified values, *I* is the number of equality constraints, $h_j(\mathbf{x})$ depicts the *j*th inequality constraint, d_j depicts pre-specified requirement values, *J* depicts the number of inequality constraints, and $\mathbf{V} = [V_1,\ldots,V_K]^T$ and $\mathbf{U} = [U_1,\ldots,U_K]^T$ denote the lower and upper bounds, respectively.

Basically, sufficient replications should be run to achieve a required accuracy in evaluating $\mathrm{E}\left[f(\mathbf{x})\right]$. Nevertheless, it is unable to execute an infinitely long simulation. There is an alternative formula to estimate the sample mean of $\mathrm{E}\left[f(\mathbf{x})\right]$.

$$\overline{f}(\mathbf{x}) = \frac{1}{L}\sum_{\ell=1}^{L} f_\ell(\mathbf{x}) \tag{5}$$

where *L* depicts the quantity of replications, and $f_\ell(\mathbf{x})$ denotes the evaluation of the ℓ th replication. The sample mean $\overline{f}(\mathbf{x})$ has a better evaluation of $\mathrm{E}\left[f(\mathbf{x})\right]$ as the number of replications increases, i.e., a large value of *L* will be more closely approximated $\mathrm{E}\left[f(\mathbf{x})\right]$.

Penalty function methods work by penalizing the infeasible solutions and converting the constrained optimization problems into unconstrained counterparts. Since both constraints of CISOP are soft ones, they are imposed by adding extra penalty terms to the cost function [34].

$$\min F(\mathbf{x}) = \overline{f}(x) + \eta \cdot \sum_{i=1}^{I} pe_i(\mathbf{x}) + \theta \cdot \sum_{j=1}^{J} pi_j(\mathbf{x}) \tag{6}$$

where η and θ depict the penalty factors of equality and inequality constraints, respectively, $F(\mathbf{x})$ denotes a penalized cost function, and $pe_i(\mathbf{x})$ and $pi_j(\mathbf{x})$ represent the quadratic penalty function.

$$pe_i(\mathbf{x}) = \begin{cases} 0, & if \ \ g_i(\mathbf{x}) = c_i, \\ \left(g_i(\mathbf{x}) - c_i\right)^2, & \text{else}, \end{cases} \quad i = 1, \ldots, I. \tag{7}$$

$$pi_j(\mathbf{x}) = \begin{cases} 0, & if \ \ h_j(\mathbf{x}) > d_j, \\ \left(h_j(\mathbf{x}) - d_j\right)^2, & \text{else}, \end{cases} \quad j = 1, \ldots, J. \tag{8}$$

The penalty factor will be a very large positive number logically. As it becomes large, the penalty for violating the constraints becomes large. The accurate estimate of (6) is obtained by $L = L_a$, where L_a denote the sufficiently large value of L. Let $F_a(\mathbf{x})$ represent the penalized cost function of $\mathbf{x}$ resulting from an accurate estimate.

2.2 Polynomial Chaos Expansion

Surrogate modeling can build a regression model through a set of available samples obtained from a design of experiments, including PCE [33], regularized minimal-energy tensor-product splines [35], extreme learning machines [36], multivariate adaptive regression splines [37], and support vector regression [38]. Among them, PCE is a method for uncertainty propagation in model-based computations under uncertainty. After obtaining a polynomial chaos expansion of a random variable, we can make a finitely parametrized and completely deterministic approximation of it. This approximation can be utilized to carry out model evaluations and obtain a mathematical representation of the model output as well. There are three distinct advantages to using the PCE: (i) it has an exact analytical expression, (ii) it can quantify the uncertainty of input parameters, and (iii) it can accurately predict the response for any set of parameters within the domain. PCE has been widely adopted in various application areas, including function approximation, prediction, curve fitting, and forecasting [33]. Accordingly, the PCE metamodel is utilized to quickly approximate a solution. Figure 1 shows the second-order PCE with three layers.

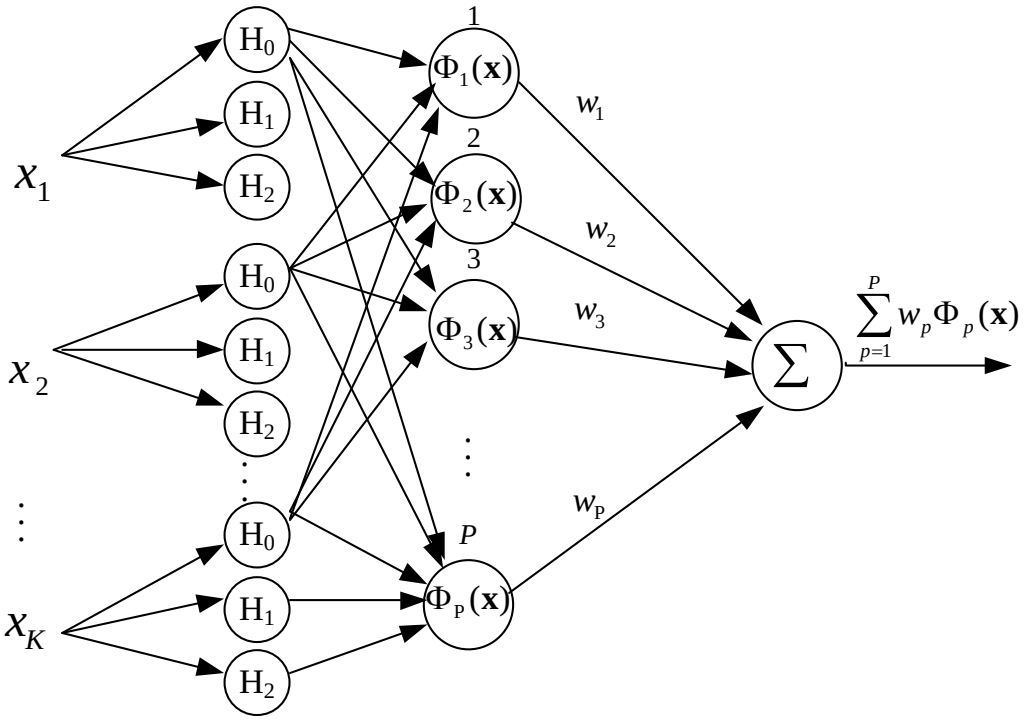

Figure. 1 A second-order PCE with three layers.

We arbitrarily select Π $\mathbf{x}$'s from design space and compute $F_a(\hat{\mathbf{x}})$ using accurate estimate, where $\hat{\mathbf{x}}=\frac{\mathbf{x}-\mu}{\sigma}$ denotes the normal standard of $\mathbf{x}$, and μ and σ denote the mean and standard deviation, respectively. These Π solutions are represented as $(\hat{\mathbf{x}}_i, F_a(\hat{\mathbf{x}}_i))$. The PCE can be approximated $F(\hat{\mathbf{x}})$ by sums of orthonormal polynomials.

$$F(\hat{\mathbf{x}}) = \sum_{p=1}^{P} w_p \Phi_p(\hat{\mathbf{x}}) \tag{9}$$

where P is the number of PCE terms; w_p denotes the expansion coefficients; and $\Phi_p(\hat{\mathbf{x}})$ depicts multivariate orthogonal polynomial basis functions, which is expressed as a product of univariate polynomials.

$$\Phi_p(\hat{x}) = \prod_{m=1}^{M} H_p\left(\hat{x}_m\right) \tag{10}$$

where M depicts the dimension of a multivariate orthogonal polynomial, which can be obtained from the input data using the Hermite polynomials $H_p(\cdot)$. These data points are

extracted from the input variables using the Hermite polynomials in the modeling process. For example, if P=2, $H_0(\hat{x})=1, H_1(\hat{x})=\hat{x}$, and $H_2(\hat{x})=\hat{x}^2-1$. The least-squares-minimization scheme is utilized to find the expansion coefficients w_p, $p=1,...,P$.

$$\begin{bmatrix} w_1 \\ \vdots \\ w_P \end{bmatrix} = \left[\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right]^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\begin{bmatrix} F_a(\hat{\mathbf{x}}_1) \\ \vdots \\ F_a(\hat{\mathbf{x}}_\Pi) \end{bmatrix} \tag{11}$$

The value of Π is larger than the value of P, i.e. $\Pi > P$. The matrix $\boldsymbol{\Phi}$ is calculated as below.

$$\boldsymbol{\Phi}=\begin{bmatrix} \Phi_1(\hat{\mathbf{x}}_1) & \Phi_2(\hat{\mathbf{x}}_1) & \cdots & \Phi_P(\hat{\mathbf{x}}_1) \\ \vdots & \vdots & & \vdots \\ \Phi_1(\hat{\mathbf{x}}_\Pi) & \Phi_2(\hat{\mathbf{x}}_\Pi) & \cdots & \Phi_P(\hat{\mathbf{x}}_\Pi) \end{bmatrix} \tag{12}$$

The PCE is trained offline to apparently reduce computational time. Since the 80:20 split draws its justification from the well-known Pareto principle, 80% of the data is for training and 20% for testing in the PCE metamodel. Once the PCE is trained, the mathematical model can be used to predict $F_a(\hat{\mathbf{x}})$ for a testing $\mathbf{x}$.

2.3 Advanced Golden Jackal Optimization

In the global search phase, we can use existing optimization approaches with the aid of the PCE to determine N excellent solutions from the entire design space. Since the GJO maintains a diverse population of solutions that can explore multiple regions of the design space simultaneously, it can meet a specific set of requirements. The GJO has many benefits, including stronger search ability, easy implementation, better stability, and few adjustment parameters.

The golden jackals are the most widely distributed in the world. They have a varied diet and populate widely diverse habitats across extensive ranges. The basic social unit of the golden jackal may be a mated pair or a family including a mated pair and its young. The jackals live in pairs and share their activities with their partners. Their behavior is highly

synchronized. Golden jackal pairs hunt, forage, and rest together. Cooperative hunting is important to jackals because hunting in pairs may be three times more successful than single hunting. Golden jackals communicate through their repertoire of calls and use a wide inventory of howls to locate one another.

The search process of GJO relies on three behaviors: search, enclose and pounce, and switch from exploration to exploitation. The behavior of searching for prey corresponds to exploration. Golden jackals adopt cooperative foraging to search an available region for large prey. The prey sometimes cannot be spotted in a specific region, and it is lost. The behavior of enclosing and pouncing on the prey corresponds to exploitation. Golden jackals will outflank the prey and encircle it until it cannot escape. They assault the prey once escape seems hopeless. The foraging hunt is guided by the male jackal, and the female jackal follows the male jackal. The escaping energy of the prey inhibits the possibility of updating the position of the golden jackal pair. Thus, it is used to determine a switch from exploration to exploitation.

The developed AGJO has two control parameters, including the escaping energy (E), and the jump strength of Levy flight (γ). The parameters E and γ are dynamically adjusted to strengthen diversification in the previous process and intensification in the subsequent process. The value of E follows an exponentially decreasing harmonic oscillation when the iteration is increased. A large E concentrates on searching for promising regions in former iterations, while a small E concentrates on looking for already-promising solutions in later iterations. The value of γ decays exponentially to intensify intensification when the iteration is increased.

The AGJO uses the following notations. Ψ is the number of golden jackals, $t_{\max}$ is the maximum iterative number, $\mathbf{x}_i^t = [x_{i,1}^t, \ldots, x_{i,K}^t]^{\mathrm{T}}$, $\mathbf{y}^t = [y_1^t, \ldots, y_K^t]^{\mathrm{T}}$ and $\mathbf{z}^t = [z_1^t, \ldots, z_K^t]^{\mathrm{T}}$ depict the positions of the ith golden jackal, male jackal, and female jackal at iteration t,

respectively. $\mathbf{xy}_i^t = [xy_{i,1}^t, \ldots, xy_{i,K}^t]^{\mathrm{T}}$ and $\mathbf{xz}_i^t = [xz_{i,1}^t, \ldots, xz_{i,K}^t]^{\mathrm{T}}$ denote the positions of the golden jackal pair corresponding to the *i*th golden jackal at iteration *t*. $E^t \in [E_{\min}, E_{\max}]$ and $\gamma^t \in [\gamma_{\min}, \gamma_{\max}]$ depict the escaping energy (*E*), and jump strength of Levy flight (γ) at iteration *t*, respectively, where $E_{\min}$, $\gamma_{\min}$ are lower bounds, and $E_{\max}$, $\gamma_{\max}$ are upper bounds.

Algorithm I: The AGJO

Step 1: Setting parameters

Define parameters to Ψ, $E_{\min}$, $E_{\max}$, $\gamma_{\min}$, $\gamma_{\max}$, and $t_{\max}$. Initialize the index variable t to zero.

Step 2: Population Initialization

Initialize a population with Ψ golden jackals.

$$\mathbf{x}_i^0 = \mathbf{V} + \mathbf{rand}[0,1] * (\mathbf{U} - \mathbf{V}), \quad i = 1, \ldots, \Psi . \tag{13}$$

where $\mathbf{V}$ and **U** denote the lower and upper bounds, respectively, and $\mathbf{rand}[0,1]$ is a vector of random numbers between zero and one.

Step 3: Determine male and female jackals

(a) Calculate $F_a(\mathbf{x}_i^t)$ of a golden jackal collaborated with PCE, $i = 1, \ldots, \Psi$.

(b) Rank the Ψ golden jackals according to their fitness from the smallest to the largest, then choose the best one as a male jackal $\mathbf{y}^t$ and the second best one as a female jackal $\mathbf{z}^t$.

Step 4: Adjust two control factors

$$E^t = \left[E_{\min} + \left(E_{\max} - E_{\min}\right) \cdot \exp\left(\ln\left(\frac{E_{\min}}{E_{\max}} \right)^2 \cdot \frac{t}{t_{\max}} \right) \right] \cdot \sin(2\pi \cdot rand[0,1]) \tag{14}$$

$$\gamma^t = \gamma_{\min} + (\gamma_{\max} - \gamma_{\min}) \cdot (1 - \exp(\frac{\gamma_{\max}}{\gamma_{\min}} \cdot (\frac{t}{t_{\max}} - 1))) \tag{15}$$

Step 5: Search for prey

If $\left|E^t\right| \geq 1$, perform searching for prey.

(1) Update the positions of the golden jackal pair corresponding to the *i*th golden jackal.

$$\mathbf{xy}_i^t = \mathbf{y}^t - E^t \cdot \left(\mathbf{y}^t - \gamma^t \cdot Levy \cdot \mathbf{x}_i^t\right), \quad i = 1,..., \Psi . \tag{16}$$

$$\mathbf{xz}_i^t = \mathbf{z}^t - E^t \cdot \left(\mathbf{z}^t - \gamma^t \cdot Levy \cdot \mathbf{x}_i^t\right), \quad i = 1,..., \Psi . \tag{17}$$

where $Levy$=$0.01 \cdot \frac{u \times \sigma}{|v|^{1/\beta}}$, u and v denote the mean and standard derivation in the Gauss distribution, σ=$\left(\frac{\Gamma(1+\beta) \cdot \sin(2\pi\beta/2)}{\Gamma\left((1+\beta)/2\right) \cdot \beta \cdot 2^{(\beta-1)/2}}\right)^{1/\beta}$, β=1.5 is a pre-designed value, and Γ is the Gamma function.

(2) Update the position of the *i*th golden jackal.

$$\mathbf{x}_i^{t+1} = \frac{\mathbf{xy}_i^t + \mathbf{xz}_i^t}{2}, \quad i = 1,..., \Psi . \tag{18}$$

When $x_{i,j}^{t+1} < V_j$, set $x_{i,j}^{t+1}$=V_j, and when $x_{i,j}^{t+1}$>U_j, set $x_{i,j}^{t+1}$=U_j.

Step 6. Enclose and pounce on prey

If $\left|E^t\right|$<1, perform enclosing and pouncing on prey.

(1) Update the positions of the golden jackal pair corresponding to the *i*th golden jackal.

$$\mathbf{xy}_i^t = \mathbf{y}^t - E^t \cdot \left(\gamma^t \cdot Levy \cdot \mathbf{y}^t - \mathbf{x}_i^t\right), \quad i = 1,..., \Psi . \tag{19}$$

$$\mathbf{xz}_i^t = \mathbf{z}^t - E^t \cdot \left(\gamma^t \cdot Levy \cdot \mathbf{z}^t - \mathbf{x}_i^t\right), \quad i = 1,..., \Psi . \tag{20}$$

(2) Update the position of the *i*th golden jackal.

$$\mathbf{x}_i^{t+1} = \frac{\mathbf{xy}_i^t + \mathbf{xz}_i^t}{2}, \quad i = 1,..., \Psi . \tag{21}$$

When $x_{i,j}^{t+1} < V_j$, set $x_{i,j}^{t+1}$=V_j, and when $x_{i,j}^{t+1}$>U_j, set $x_{i,j}^{t+1}$=U_j.

Step 7: Terminating condition

If $t \geq t_{\max}$, stop; else, set $t = t+1$ and repeat from Step 3.

The AGJO terminates after $t_{\max}$ iterations have been performed. When the AGJO is stopped, the Ψ golden jackals are ranked based on their fitness. Although the AGJO is developed for continuous variables, a real value can be transformed to the greatest integer less than or equal to it by the floor function $s_{i,k}^{t_{\max}} = \lfloor x_{i,k}^{t_{\max}} \rfloor$, where $x_{i,k}^{t_{\max}} \in \mathbb{R}$ and $s_{i,k}^{t_{\max}} \in \mathbb{Z}$. Consequently, the former N golden jackals are picked up to construct the excellent subset.

2.4 Modified Optimal Computing Budget Allocation

The optimal computing budget allocation (OCBA) has been shown to be very efficient, controllable, and robust when compared to other ranking and selection methods. To improve the effectiveness of the OCBA, the MOCBA is developed to select the optimal solution under the restriction of an extremely small computing budget. All replications are executed in the allocation procedure to yield the mean and variance for each design in the OCBA. The MOCBA just needs to perform incremental replications in the allocation procedure to yield the mean and variance for each design. Thus, MOCBA proposes a way of asymptotically optimally allocating the extremely small computing budget among competing designs.

Let L_0 denote the initial replications assigned to each design, L_n indicate the replications allocated to the nth design, and C_ℓ indicate the limited computing budget. An additional computing budget, Δ, is added in each allocation procedure. The value of Δ is obtained through experimentation. The target of the MOCBA is to maximize the probability of correct selection under $L_1 + L_2 + \cdots + L_N = C_\ell$ to allocate C_ℓ wisely to $L_1, \ldots, L_N$. The limited computing budget C_ℓ is obtained by $C_\ell = \frac{N \cdot L_a}{\tau}$, where L_a denote the replications of an accurate estimate, and τ denotes a speeding up of parameters [39].

Algorithm II: The MOCBA

Step 1. Configure L_0, $l = 0$, $L_n^l = L_0$, $n = 1, ..., N$, and compute the limited computing

budget $C_\ell = \dfrac{N \cdot L_a}{\tau}$.

Step 2. Add Δ to $\sum_{n=1}^{N} L_n^l$, and calculate the required replications.

$$L_j^{l+1} = (\sum_{n=1}^{N} L_n^l + \Delta) \cdot \theta_j^l / (\theta_b^l + \sum_{n=1, n\neq b}^{N} \theta_n^l) \tag{22}$$

$$L_b^{l+1} = \frac{\theta_b^l}{\theta_j^l} \cdot L_j^{l+1} \tag{23}$$

$$L_n^{l+1} = \frac{\theta_n^l}{\theta_j^l} \cdot L_j^{l+1} \tag{24}$$

where $\dfrac{\theta_n^l}{\theta_j^l} = \left(\dfrac{\delta_n^l \cdot (\bar{f}_b^l - \bar{f}_j^l)}{\delta_j^l \cdot (\bar{f}_b^l - \bar{f}_n^l)} \right)^2$, $\theta_b^l = \delta_b^l \sqrt{\sum_{n=1, n\neq b}^{N} (\dfrac{\theta_n^l}{\delta_n^l})^2}$, $\bar{f}_n^l = \dfrac{1}{L_n^l} \sum_{k=1}^{L_n^l} f_k(\mathbf{x}_n)$,

$\delta_n^l = \sqrt{\dfrac{1}{L_n^l} \sum_{k=1}^{L_n^l} \left(f_k(\mathbf{x}_n) - \bar{f}_n^l \right)^2}$ for all $n \neq j \neq b, b = \arg\min_n \bar{f}_n^l$, $\mathbf{x}_n$ denotes the n th design, and $f_k(\mathbf{x}_n)$ represents the objective function of $\mathbf{x}_n$ at the k th replication.

Step 3. Perform $\max[0, L_n^{l+1} - L_n^l]$ incremental replications to the n th design, and calculate the incremental mean ($\hat{f}_n^{l+1}$) and incremental standard deviation ($\hat{\delta}_n^{l+1}$).

$$\hat{f}_n^{l+1} = \frac{1}{(L_n^{l+1} - L_n^l)} \sum_{k=L_n^l+1}^{L_n^{l+1}} f_k(\mathbf{x}_n) \tag{25}$$

$$\hat{\delta}_n^{l+1} = \sqrt{\frac{1}{(L_n^{l+1} - L_n^l)} \sum_{k=L_n^l+1}^{L_n^{l+1}} \left(f_k(\mathbf{x}_n) - \hat{f}_n^{l+1} \right)^2} \tag{26}$$

Step 4. Calculate the complete mean ($\bar{f}_n^{l+1}$) and complete standard deviation (δ_n^{l+1}) of the n th design for entire replications.

$$\bar{f}_n^{l+1} = \frac{1}{L_n^{l+1}} \left(L_n^l \cdot \bar{f}_n^l + (L_n^{l+1} - L_n^l) \cdot \hat{f}_n^{l+1} \right) \tag{27}$$

$$\delta_n^{l+1}=\sqrt{\frac{1}{(L_n^{l+1}-1)}\cdot\left(L_n^l\left(\bar{f}_n^l\right)^2+(L_n^l-1)\left(\delta_n^l\right)^2+(L_n^{l+1}-L_n^l)\left(\hat{f}_n^{l+1}\right)^2+(L_n^{l+1}-L_n^l-1)\left(\hat{\delta}_n^{l+1}\right)^2-L_n^{l+1}\left(\hat{f}_n^{l+1}\right)^2\right)} \quad (28)$$

Step 5. If $\sum_{n=1}^{N} L_n^l \geq C_\ell$, stop and choose the best $\mathbf{x}^*$ such that objective value is minimal; else, set $l=l+1$ and repeat from Step 2.

2.5 The GJOO Algorithm

Figure 2 shows the flowchart of the GJOO algorithm. First of all, the PCE is used to assess a solution more readily in the block of a metamodel. Secondly, the AGJO cooperates with the PCE metamodel to find *N* excellent solutions from the entire design space in the block of global search. At last, the MOCBA is utilized to select a distinguished solution among the *N* excellent solutions in the block of ranking and selection. The computational complexity of GJOO with Ψ jackals depends on the initialization and updating of golden jackals. The computational complexity is $O(\Psi)$ in the initialization. In the updating phase, the computational complexity is $O(t_{\max}\cdot\Psi)+O(t_{\max}\cdot\Psi\cdot K)$, which comprises updating the location of all jackals and searching for the best location, where $t_{\max}$ denotes the maximum iterative number and *K* is the dimension of CISOP. Thus, the computational complexity of the GJOO is $O\left(\left[1+t_{\max}\cdot(1+K)\right]\cdot\Psi\right)$.

Algorithm III: The GJOO

Step 1: Define the values of Ψ, $E_{\min}$, $E_{\max}$, $\gamma_{\min}$, $\gamma_{\max}$, $t_{\max}$, N, L_a, L_0, and Δ.

Step 2: Arbitrarily sample Π $\mathbf{x}$'s from the design space, calculate $F_a(\mathbf{x})$ using accurate estimate, and construct the PCE through these Π samples.

Step 3: Yield Ψ $\mathbf{x}$'s to be the initial population, then adopt the AGJO algorithm for those golden jackals that collaborated with PCE. After the AGJO algorithm stops, sort the Ψ $\mathbf{x}$'s according to fitness from the smallest to the largest, and pick up the former N $\mathbf{x}$'s to constitute the excellent subset.

Step 4: Adopt the MOCBA algorithm for the N excellent solutions and find the optimum $\mathbf{x}^*$, which is the distinguished solution.

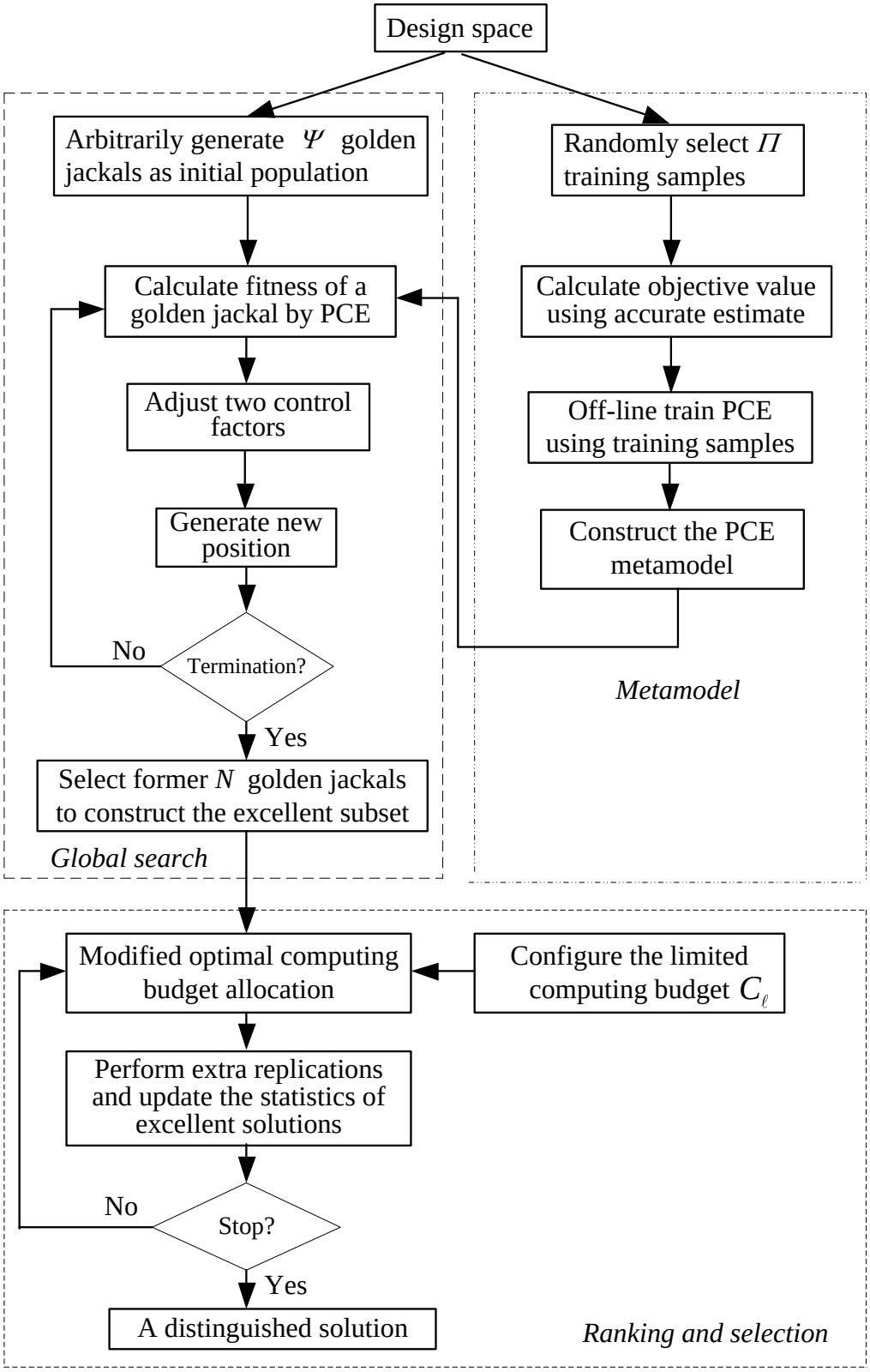

Figure. 2 Flowchart of the GJOO algorithm.

3. Shipping/Receiving Docks Optimization of Container Freight Station

3.1 Container Freight Station in Air Cargo

Affected by COVID-19, the global manufacturing supply chain has been disrupted, and shipping vessels have blocked ports. The shortage of cargo containers has driven the demand for air cargo. The air cargo terminal is a necessary gateway for the import and export of goods. When the air cargo terminal is limited by the airport area, the optimization of the shipping/receiving docks can improve efficiency and maintain the growth momentum of air cargo [40]. Figure 3 shows the shipping/receiving docks of a container freight station. Since import and export cargo is picked up and delivered by trucks at the docks, the shipping/receiving docks are key resources at a certain air cargo terminal. There are four major categories of cargo: pallet bulk, general bulk, perishable cargo, and prepacked cargo. Different categories of cargo require different operations and utilize different material handling systems. Accordingly, the terminal must determine how to assign the shipping/receiving docks to the four categories. It is particularly important during the peak season, such as the month before Christmas and New Year's, because the terminal is operating at default capacity.

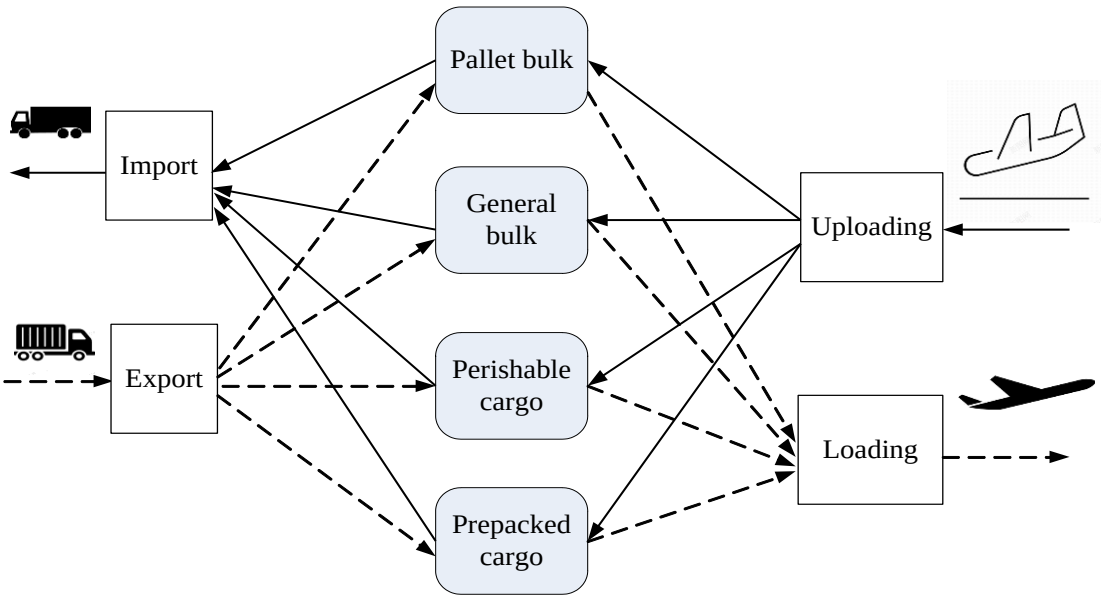

Figure 3. Shipping/receiving docks of a container freight station.

The long-run average truck waiting time is usually regarded as a performance measure, which is defined as the time waiting to get to a dock plus the time required to complete the service. Five common approaches to helping find the optimal number of docks are the exhaustive search, greedy method, approximate dynamic programming technique, branch and bound method, and heuristic algorithms. An exhaustive search is a brute-force scheme that systematically enumerates all possible solutions to the problem. The greedy method solves a problem by selecting the most appropriate option based on the current situation [41]. The approximate dynamic programming technique integrates function approximation and simulation to appease the curse-of-dimensionality involved with the dynamic programming method [42]. The branch-and-bound scheme adopts a divide-and-conquer scheme to divide the design space into subproblems, and each sub-problem is optimized independently. If the worst-case scenario occurs, we need to search all the design spaces. Although heuristic algorithms can quickly generate a solution with acceptable quality, one of the main drawbacks is the high probability of being trapped in local optima.

Instead of handling an approximate mathematical model, the considered problem can be modeled as a CISOP. Therefore, the shipping/receiving docks optimization problem of a container freight station in air cargo is first formulated as a CISOP. Next, the GJOO is utilized to determine the number of docks needed to minimize the long-run average truck waiting time while ensuring the stability of the queues.

3.2 Mathematical Formulation

Consider an air cargo terminal that has *M* docks. The arrival processes follow Poisson processes with a stationary arrival rate λ_j, $j = 1,...,4$. Service times for four cargo categories follow an exponential distribution with a stationary rate μ_j, $j = 1,...,4$. Let x_i, $i = 1,...,4$, depict the number of docks allocated to the pallet bulk, general bulk, perishable

cargo, and prepacked cargo, respectively, so that $\sum_{i=1}^{4} x_i = M$. In addition, we require that $x_j > \lambda_j / \mu_j$, $j = 1, \ldots, 4$ to guarantee the stability of the queues.

Now, the shipping/receiving docks optimization problem of a container freight station in air cargo is formulated as a CISOP.

$$\min \quad \mathrm{E}\left[f(\mathbf{x}) \right] \tag{29}$$

$$\text{subject to} \quad \sum_{i=1}^{4} x_i = M , \tag{30}$$

$$x_j > \lambda_j / \mu_j , \; j = 1, \ldots, 4, \tag{31}$$

$$\mathbf{V} \le \mathbf{x} \le \mathbf{U} . \tag{32}$$

where $\mathbf{x} = [x_1, \ldots, x_4]^T$ indicates a solution, $x_1 \sim x_4$ depict the number of docks, M denotes the total number of docks, $E[f(\mathbf{x})]$ represents the long-run average truck waiting time, λ_j is the arrival rate of cargo, μ_j is the service rate of cargo, and $\mathbf{V} = [1,1,1,1]^T$, $\mathbf{U} = [M, M, M, M]^T$ represent the lower and upper bounds, respectively.

The purpose of the CISOP is to determine the optimal number of docks $\mathbf{x}^*$ for minimizing the long-run average waiting time, subject to the limit of docks, stability constraints, and integrality conditions. The sample mean $\bar{f}(\mathbf{x})$ is used to approximate the expected value $E[f(\mathbf{x})]$.

$$\bar{f}(\mathbf{x}) = \frac{1}{L} \sum_{\ell=1}^{L} f_\ell(\mathbf{x}) \tag{33}$$

where L denotes the quantity of replications, and $f_\ell(\mathbf{x})$ denote the estimate of the ℓ th replication. Since two constraints are soft ones, the constrained optimization problems are transformed into unconstrained counterparts.

$$\min \quad F(\mathbf{x}) = \bar{f}(\mathbf{x}) + \eta \cdot \left(\sum_{i=1}^{4} x_i - M \right)^2 + \theta \cdot \sum_{j=1}^{4} pi_j(\mathbf{x}) \tag{34}$$

where η and θ depict the penalty factor, $F(\mathbf{x})$ is the penalized cost function, and $pi_j(\mathbf{x})$ is defined as follows.

$$pi_j(\mathbf{x}) = \begin{cases} 0, \ if \ \ x_j > \lambda_j / \mu_j , \\ \left(\lambda_j / \mu_j - x_j \right)^2, \ else, \end{cases}, \quad j = 1, ..., 4 . \tag{35}$$

Let L_a indicate the sufficiently large value of L, and the accurate estimate of (34) is defined as $L = L_a$. Let $F_a(\mathbf{x})$ denote the penalized cost function of $\mathbf{x}$ resulted from an accurate estimate.

3.3 Application of the GJOO Method

3.3.1 Train the PCE metamodel

Four steps were adopted to constitute the PCE metamodel to assess a solution. (i) Randomly generate Π $\mathbf{x}$'s and evaluate $F_a(\mathbf{x})$ through an accurate estimate, then denote these Π samples and their estimates as $\mathbf{x}_i$ and $F_a(\mathbf{x}_i)$, respectively. (ii) Define the quantity of PCE terms, i.e., P=2. (iii) Calculate the matrix $\mathbf{\Phi}$. (iv) Apply the least-squares-minimization scheme to find the coefficients w_p.

3.3.2 Constitute the excellent subset

Firstly, Ψ golden jackals were arbitrarily selected to construct the initial population. The fitness of a golden jackal was estimated by the PCE metamodel. Once the AGJO was terminated, the Ψ golden jackals were ranked based on their fitness. The prior N golden jackals were picked up to constitute the excellent subset.

3.3.3 Determine the Distinguished Solution

The MOCBA algorithm was applied to determine a distinguished solution from the N excellent solutions. Ref. [39] revealed that an appreciated value of Δ is greater than 10% of *N* but less than 100, and an appreciated value of L_0 is between 5 and 20.

4. Numerical experiment and result analysis

4.1. Test Example

A test example of an air cargo terminal adapted from [43] is used to verify the GJOO method. The total number of docks M=115. The cargo arrival rates follow a stationary Poisson process based on Table 1. Service times for four cargo categories follow an exponential distribution with a stationary rate based on Table 1. Every replication started from an empty system and had a warm-up simulation time of 100 hours. Then, we simulated for 500 hours after the warm-up period. Time was measured according to the quantity of replications that were executed.

Table 1 Arrival Rates and Average Service Times.

Type of Cargo	Arrival Rate (1/minute)	Average Service Time (minutes)
pallet bulk	$\lambda_1 = 52.8/60$	$1/\mu_1 = 67$
general bulk	$\lambda_2 = 11.7/60$	$1/\mu_2 = 46$
Perishable cargo	$\lambda_3 = 13/60$	$1/\mu_3 = 92$
prepacked cargo	$\lambda_4 = 22.5/60$	$1/\mu_4 = 34$

There are 9604 arbitrarily chosen solutions to train the PCE. The value of Π=9604 was calculated through the sampling size formula with a confidence level of 95% as well as a confidence interval of 1%.

The penalty factors were η=10 and θ=20, which were obtained through hand-tuned experiments. The lower and upper bounds were $\mathbf{V} = [1,1,1,1]^T$ and $\mathbf{U} = [115,115,115,115]^T$, respectively. Therefore, the size of the design space is 115^4. The parameters utilized in AGJO were $E_{\min}$=0.1, $E_{\max}$=4, $\gamma_{\min}$=0.05, $\gamma_{\max}$=0.4 $t_{\max} = 300$, and $\Psi = 100$. Figure 4 displays the curves of two control factors, E and γ, over 300 iterations. The GJOO was simulated with four cases of N, which were N=40, 30, 20, and 10, to investigate the effect of N. The

parameters used in MOCBA were L_0 =20, Δ =10 and $L_a = 10^4$. The speeding up of parameters τ that correspond to *N*=40, 30, 20, and 10 are 10.7, 8.4, 6.1, and 3.3 [39], respectively. Therefore, the limited computing budgets C_ℓ were 37383, 35714, 32787, and 30303 for *N*=40, 30, 20, and 10, respectively.

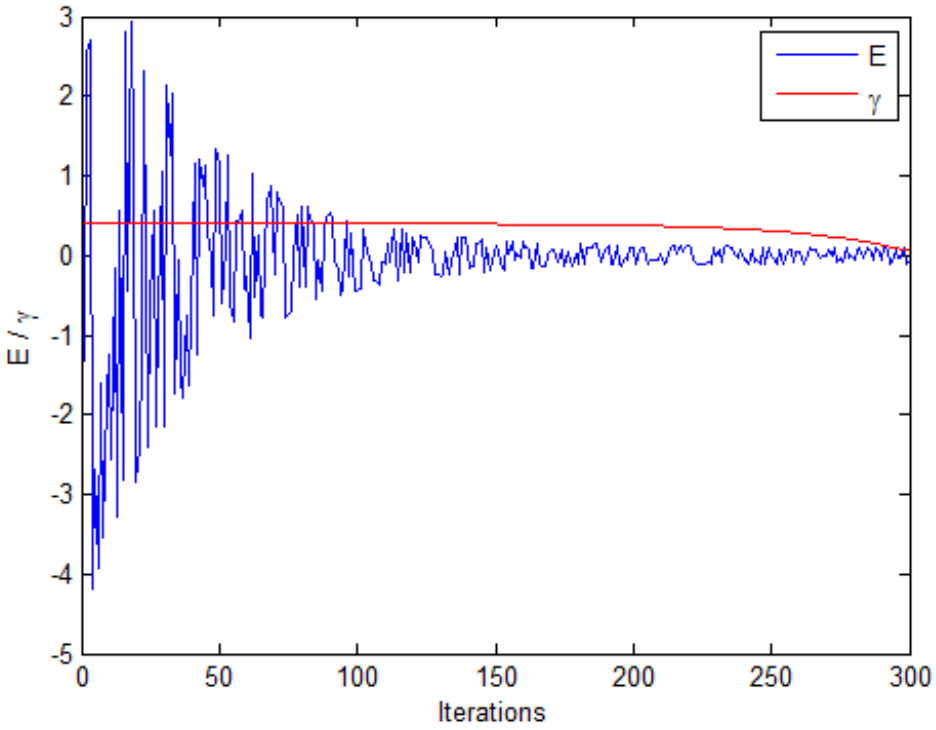

Figure. 4 Variations of E and γ over iterations.

Table 2 demonstrates the distinguished solution $\mathbf{x}^*$, cost $E[f(\mathbf{x}^*)]$, and CPU times of four cases. For case *N*=40, the optimal assignment spends 10.25 minutes, 64 docks for pallet bulk, 12 docks for general bulk, 23 docks for perishable cargo, and 16 docks for prepacked cargo. The CPU time consumed was smaller than two minutes of four cases, which illustrates that the GJOO can meet the real-time needs. The GJOO algorithm was implemented on MATLAB R2019a software on Windows 10, which was performed on a personal computer with an Intel Core i7-8550U at 1.80GHz and 32 GB of RAM. Interested readers may refer to Ref. [44] for the source code of the GJOO approach.

Table 2. The distinguished solution, cost, and CPU times of four cases.

N	$\mathbf{x}^*$	$E[f(\mathbf{x}^*)]$ (min.)	CPU times (sec.)
40	$[64,12,23,16]^T$	10.25	118.32
30	$[64,11,24,16]^T$	10.96	110.56
20	$[65,11,23,16]^T$	11.12	105.73
10	$[66,11,23,15]^T$	12.65	97.62

4.2 Result Analysis

The GJOO approach was compared to five meta-heuristic algorithms for case N=40: African vultures optimization algorithm (AVOA) [45], ant colony optimization (ACO) [46], clonal selection algorithm (CSA) [47], multi-cohort whale optimization algorithm (MCWOA) [48], and sine-cosine spotted hyena-based chimp optimization algorithm (SSC) [49]. We have adopted a population size of 100 with a maximum iterative number of 300 to ensure the same setting as the GJOO algorithm. Other algorithm-specific parameter settings for all meta-heuristic algorithms are summarized in Table 3.

The five meta-heuristic methods use the accurate estimate compute the objective value. Due to randomness, 30 trials were carried out to inspect the validity and reliability. Since the five meta-heuristic algorithms consume more computing time to obtain the optimum, the optimization processes stopped when they had spent one hour of computing time. Table 4 demonstrates the statistical analysis and average CPU times over 30 trials for six methods. The averages of the elite objective value resulted from AVOA, ACO, CSA, MCWOA, and SSC were still 30.81%, 30.07%, 32.65%, 24.68%, and 26.92% larger than those obtained by GJOO, respectively. Simulation results reveal that the GJOO outperforms five meta-heuristic algorithms.

Table 3. Parameter setting values for five meta-heuristic methods.

Algorithms	Parameters	Value
African vultures optimization algorithm	Probability of selections in exploration phase P_1	0.6
	Probability of selections in the first part of exploitation phase P_2	0.4
	Probability of selections in the second part of exploitation phase P_3	0.6
	Probability toward the best solutions for group L_1	0.8
	Probability toward the best solutions for group L_2	0.2
	Parameter disrupts the exploration and exploitation phases *w*	2.5
Ant colony optimization	Initial pheromone	0.1
	Volatile factor for global pheromone	0.3
	Evaporation rate for local pheromone	0.5
	Relative importance of pheromones	1
	Probability for selecting exploitation and exploration	0.9
Clonal selection algorithm	Strength of mutation	10
	Receptor editing rate	0.5
Multi-cohort whale optimization algorithm	Spiral factor *b*	1
	Convergence constant *a*	[2,0]
Sine-cosine and Spotted hyena-based chimp optimization algorithm	Encircling behavior *h*	[5,0]
	Chimp coefficient *l*	[2.5,0]

Table 4. Statistic analysis and average CPU times of six algorithms.

Algorithms	Min.	Max.	AEOV†	$\frac{\text{AEOV}^{\dagger}-*^{\S}}{*}\times 100\%$	S.D.	S.E.M.	Average rank percentage	Average CPU time(min.)
GJOO	10.24	10.34	10.29	0	0.02	0.0037	0.02%	1.96
AVOA with accurate estimate	13.07	13.85	13.46	30.81%	0.12	0.0219	4.83%	59.97
ACO with accurate estimate	12.99	13.77	13.38	30.07%	0.13	0.0237	4.54%	60.02
CSA with accurate estimate	13.22	14.11	13.65	32.65%	0.14	0.0256	6.68%	59.93
MCWOA with accurate estimate	12.62	13.15	12.83	24.68%	0.09	0.0164	2.91%	59.99
SSC with accurate estimate	12.71	13.46	13.06	26.92%	0.11	0.0201	3.32%	59.95

†AEOV: average of the elite objective value

§ ∗: AEOV resulting from GJOO

At last, a percentage analysis concerning rank was performed to demonstrate the order of a distinguished solution in the design space. Since it is difficult to determine the order of all solutions, a sampling subset, Ω, is used to reflect the characteristics of the entire design space. The percentage concerning rank of a distinguished solution is defined as $\frac{O}{|\Omega|}\times 100\%$, where O denotes the order of a distinguished solution in Ω. In general, 13572 samples were arbitrarily chosen from the entire design space to constitute the sampling subset. The objective values of all samples were obtained by accurate estimation. The value of $|\Omega|$=13572 was obtained using a confidence level of 98% and a confidence interval of 1%. Table 4 also demonstrates the average percentages concerning rank obtained by six algorithms. The standard error of the mean (S.E.M.) obtained by GJOO was 0.0037. This tiny S.E.M. reveals that most of the distinguished solutions obtained by the GJOO are relatively near the optimum for 30 trials.

The convergence status of each method can be seen in detail from the convergence curve. Therefore, we analyze the convergence status of the six methods that collaborated with the PCE metamodel by drawing the convergence curve in the global search. Figure 5 describes the convergence results of six methods for the first trial. The horizontal axis indicates the maximum iteration times, and the vertical axis represents the objective value in the global search phase. Results demonstrate that AGJO has improved significantly in the global search phase. For the other five methods, MCWOA has the better convergence, followed by SSC, AVOA, CSA, and ACO.

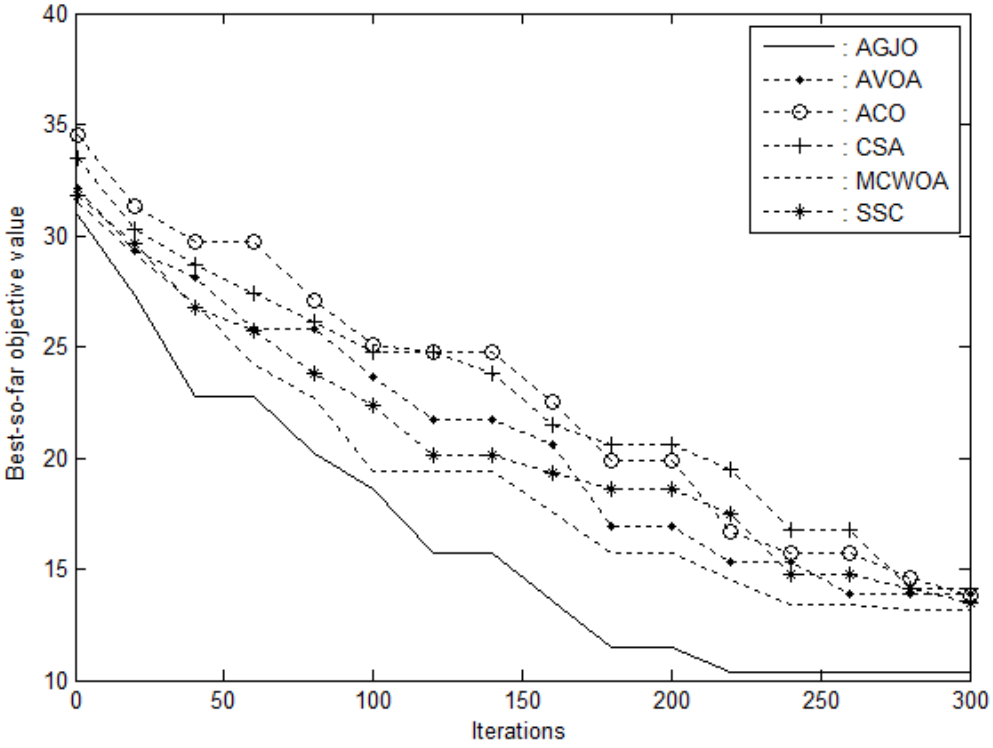

Figure 5. Convergence curve of the six methods in the global search phase.

To analyze the significance of results, the Wilcoxon rank sum test is performed at the 5% significance level [50]. Through statistical analysis of the data of two groups, the *p*-value and *h*-value are adopted as indicators to assess whether each method has statistical significance. The *p*-value represents the level of marginal significance within a statistical hypothesis test. The value $h = 1$ represents a rejection of the null hypothesis, and $h = 0$ indicates a failure to reject the null hypothesis. The data of two groups are significantly different if p-value<0.05 and $h =1$ are obtained. The comparison results of the Wilcoxon rank sum test for AGJO versus five meta-heuristic methods are shown in Table 5. The AGJO is significantly different from five meta-heuristic algorithms.

Table 5. Comparison results of the Wilcoxon rank sum test.

Value	AGJO vs. AVOA	AGJO vs. ACO	AGJO vs. CSA	AGJO vs. MCWOA	AGJO vs. SSC
p-value	0.0013	2.8321×10^{-4}	1.2493×10^{-5}	0.0162	0.0049
h-value	1	1	1	1	1

5. Conclusions and Further research

To solve the CISOP in a reasonable time, an algorithm that uses GJO, assisted by OO, was presented. The GJOO consists of three phases: the metamodel, global search, and ranking and selection. The PCE metamodel was used to quickly evaluate a solution. The GJOO used the AGJO for global search and the MOCBA for ranking and selection. The GJOO was applied to the shipping/receiving docks optimization problem of a container freight station in air cargo, which was modeled as a CISOP. A practical example of an air cargo terminal was adopted to inspect the GJOO approach. The CPU time consumed was smaller than two minutes, which reveals that the GJOO can meet the real-time needs. The GJOO was compared to five meta-heuristic algorithms: AVOA, ACO, CSA, MCWOA, and SSC. Simulation results showed that most of the distinguished solutions obtained by the GJOO were relatively near the optimum for 30 trials. A Wilcoxon rank sum test is carried out to inspect the significance of AGJO from a statistical point of view. Test results show that AGJO is significantly different from five meta-heuristic algorithms.

The application of OO is not restricted to the shipping/receiving docks optimization problem of a container freight station in air cargo. OO can be adopted to assist novel meta-heuristic methods, such as the sine cosine algorithm, the memory-based hybrid dragonfly algorithm, the arithmetic optimization algorithm, the artificial ecosystem-based optimization algorithm, and the Aquila optimizer. Further research will extend OO to resolve more complex optimization problems involving stochastic constraints, including conditional value-at-risk optimization problems, and risk-averse stochastic optimization problems.

References

[1] T. Giovannelli, G. Liuzzi, S. Lucidi, F. Rinaldi, “Derivative-free methods for mixed-integer nonsmooth constrained optimization”, Computational Optimization and Applications, Vol. 82, No. 2, pp. 293-327, Jun. 2022.

[2] D. Garmatter, M. Porcelli, F. Rinaldid, M. Stoll, “Improved penalty algorithm for mixed integer PDE constrained optimization problems”, Computers & Mathematics with Applications, Vol. 116, pp. 2-14, Jun. 2022.

[3] D. Bertsimas, R. Cory-Wright, J. Pauphilet, “A unified approach to mixed-integer optimization problems with logical constraints”, SIAM Journal on Optimization, Vol. 31, No. 3, pp. 2340-2367, 2021.

[4] C. Geiersbach, E. Loayza-Romero, K. Welker, “Stochastic approximation for optimization in shape spaces”, SIAM Journal on Optimization, Vol. 31, No. 1, pp.348-376, 2021.

[5] X.J. Zhou, X.Y. Wang, T.W. Huang, C.H. Yang, “Hybrid intelligence assisted sample average approximation method for chance constrained dynamic optimization”, IEEE Transactions on Industrial Informatics, Vol. 17, No. 9, pp. 6409-6418, Sep. 2021.

[6] Q.B. Zhang, S.X. Yang, M. Liu, J.X. Liu, L. Jiang, “A new crossover mechanism for genetic algorithms for Steiner tree optimization”, IEEE Transactions on Cybernetics, Vol. 52, No. 5, pp. 3147-3158, May 2022.

[7] C.L. Yu, N. Lahrichi, A. Matta, “Optimal budget allocation policy for tabu search in stochastic simulation optimization”, Computers & Operations Research, Vol. 150, id. 106046, Feb. 2023.

[8] D.L. Cheng, “Water allocation optimization and environmental planning with simulated annealing algorithms”, Mathematical Problems in Engineering, Vol. 2022, id. 2281856, May 2022.

[9] J. Tang, G. Liu, Q.T. Pan, “A review on representative swarm intelligence algorithms for

solving optimization problems: applications and trends", IEEE-CAA Journal of Automatica Sinica, Vol. 8, No. 10, pp. 1627-1643, Oct. 2021.

[10] N. Chopraa, M.M. Ansarib, "Golden jackal optimization: A novel nature-inspired optimizer for engineering applications", Expert Systems with Applications, Vol. 198, id. 116924, Jul. 2022.

[11] E.H. Houssein, D.A. Abdelkareem, M.M. Emam, M.A. Hameed, M. Younan, "An efficient image segmentation method for skin cancer imaging using improved golden jackal optimization algorithm", Computers in Biology and Medicine, Vol. 149, id. 106075, Oct. 2022.

[12] M.A. Hameed, O.A. Abdel-Aleem, M. Hassaballah, "A secure data hiding approach based on least-significant-bit and nature-inspired optimization techniques", Journal of Ambient Intelligence and Humanized Computing, Vol. 14, No. 5, pp. 4639-4657, May 2023.

[13] M. Hassaballah, M. A. Hameed, A. I. Awad, K. Muhammad, "A Novel Image steganography method for industrial internet of things security", IEEE Transactions on Industrial Informatics, Vol. 17, No. 11, pp. 7743-7751, Nov. 2021.

[14] M. Braik, A. Hammouri, J. Atwan, M.A.A. Al-Betar, M.A Awadallah, "White Shark Optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems", Knowledge-Based Systems, Vol. 243, id.108457, May 2022.

[15] B. Abdollahzadeh, F.S. Gharehchopogh, S. Mirjalili, "African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems", Computers & Industrial Engineering, Vol. 158, id. 107408, Aug. 2021.

[16] G. Dhiman, "SSC: A hybrid nature-inspired meta-heuristic optimization algorithm for engineering applications", Knowledge-Based Systems, Vol. 222, id. 106926, Jun. 2021.

[17] Y.X. Jiang, Q. Wu, S.K. Zhu, L.K. Zhang, "Orca predation algorithm: A novel

bio-inspired algorithm for global optimization problems", Expert Systems with Applications, Vol. 188, id. 116026, Feb. 2022.

[18] F.A. Hashim, A.G. Hussien, "Snake optimizer: A novel meta-heuristic optimization algorithm", Knowledge-Based Systems, Vol. 242, id. 108320, Apr. 2022.

[19] H.M. Jia, X.X. Peng, C.B. Lang, "Remora optimization algorithm", Expert Systems with Applications, Vol. 185, id. 115665, Dec. 2021.

[20] M. Dehghani, Z. Montazeri, E. Trojovska, P. Trojovsky, "Coati optimization algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems", Knowledge-Based Systems, Vol. 259, id. 110011, Jan. 2023.

[21] W.G. Zhao, L.Y. Wang, S. Mirjalili, "Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications", Computer Methods in Applied Mechanics and Engineering, Vol. 388, id. 114194, Jan. 2022.

[22] D. Polap, M. Wozniak, "Red fox optimization algorithm", Expert Systems with Applications, Vol. 166, id. 114107, Mar. 2021.

[23] S, Kaur, L.K. Awasthi, A.L. Sangal, G. Dhiman, "Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization", Engineering Applications of Artificial Intelligence, Vol. 90, id. 103541, Apr. 2020.

[24] L. Abualigah, M. Abd Elaziz, P. Sumari, Z.W. Geem, A.H. Gandomi, "Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer", Expert Systems with Applications, Vol. 191, id. 116158, Apr. 2022.

[25] M. Dehghani, P. Trojovsky, "Serval optimization algorithm: A new bio-inspired approach for solving optimization problems", Biomimetics, Vol. 7, No. 4, id. 204, Dec. 2022.

[26] S. Rajmohan, E. Elakkiya, S.R. Sreeja, "Multi-cohort whale optimization with search space tightening for engineering optimization problems", Neural Computing & Applications, Vol. 35, No. 12, pp. 8967-8986, Apr. 2023.

[27] H.L. Minh, T. Sang-To, G. Theraulaz, M.A. Wahab, T. Cuong-Le, “Termite life cycle optimizer”, Expert Systems with Applications, Vol. 213, id. 119211, Mar. 2023.

[28] Y.C. Ho, Q.C. Zhao, Q.S. Multi-cohort whale optimization with search space tightening for engineering optimization problems Jia, *Ordinal optimization: Soft optimization for hard problems*. New York: Springer-Verlag, 2007.

[29] S.C. Horng, C.T. Lee, “Integration of ordinal optimization with ant lion optimization for solving the computationally expensive simulation optimization problems”, Applied Sciences, Vol. 11, No. 1, id. 136, Jan. 2021.

[30] S.C. Horng, S.S. Lin, “Incorporate seagull optimization into ordinal optimization for solving the constrained binary simulation optimization problems”, The Journal of Supercomputing, Vol. 79, No. 5, pp. 5730-5758, Mar. 2023.

[31] S.C. Horng, S.S. Lin, “Ordinal optimization to optimize the job-shop scheduling under uncertain processing times”, Arabian Journal for Science and Engineering, Vol. 47, No. 8, pp. 9659 - 9671, Aug. 2022.

[32] S.C. Horng, S.S. Lin, “Improved beluga whale optimization for solving the simulation optimization problems with stochastic constraints”, Mathematics, Vol. 11, No. 8, id.1854, Apr. 2023.

[33] Y. Liu, G. Zhao, G. Li, W.X. He, C.T. Zhong, “Analytical robust design optimization based on a hybrid surrogate model by combining polynomial chaos expansion and Gaussian kernel”, Structural and Multidisciplinary Optimization, Vol. 65, No. 11, id. 335, Nov. 2022.

[34] R. Estrin, M.P. Friedlander, D. Orban, M.A. Saunders, “Implementing a smooth exact penalty function for equality-constrained nonlinear optimization”, SIAM Journal on Scientific Computing, Vol. 42, No. 3, pp. A1809-A1835, 2020.

[35] S.H. Huang, K. Mahmud, C.J. Chen, “Meaningful trend in climate time series: A

discussion based on linear and smoothing techniques for drought analysis in Taiwan", Atmosphere, Vol. 13, No. 3, id. 444, Mar. 2022.

[36] W.D. Zou, Y.Q. Xia, W.P. Cao, "Back-propagation extreme learning machine", Soft Computing, Vol. 26, No. 18, pp.9179-9188, Sep. 2022.

[37] Q.L. Zuo, "Settlement prediction of the piles socketed into rock using multivariate adaptive regression splines", Journal of Applied Science and Engineering, Vol. 26, No. 1, pp. 111-119, 2023.

[38] T. Uemoto, K. Naito, "Support vector regression with penalized likelihood", Computational Statistics & Data Analysis, Vol. 174, id. 107522, Oct. 2022.

[39] C.H. Chen, L.H. Lee, *Stochastic Simulation Optimization: An Optimal Computing Budget Allocation*, World Scientific, New Jersey, 2010.

[40] H.Q. Wang, H.D. Haasis, M.H. Su, J.H. Wei, X.B. Xu, S.J. Wen, J.T. Li, W.X. Yue, "Improved artificial bee colony algorithm for air freight station scheduling", Mathematical Biosciences and Engineering, Vol. 19, No. 12, pp.13007-13027, 2022.

[41] H.X. Wang, F. Xie, J. Li, F. Miu, "Modelling, simulation and optimisation of medical enterprise warehousing process based on FlexSim model and greedy algorithm", International Journal of Bio-Inspired Computation, Vol. 19, No. 1, pp. pp.59-66, 2022.

[42] Y.Z. Meng, R.R. Chen, T.H. Deng, "Two-stage robust optimization of power cost minimization problem in gunbarrel natural gas networks by approximate dynamic programming", Petroleum Science, Vol. 19, No. 5, pp. 2497-2517, Oct. 2022.

[43] L.J. Hong, B.L. Nelson, "A framework for locally convergent random-search algorithms for discrete optimization via simulation", ACM Transactions on Modeling and Computer Simulation, Vol. 17, No. 4, id. 19, Sep. 2007.

[44] S.C. Horng and S.S. Lin, “Advanced golden jackal optimization for solving the constrained integer stochastic optimization problems”, Online Appendix, https://www.cyut.edu.tw/~schong/eng/technical.htm, Jun. 2023.

[45] B. Abdollahzadeh, F. S. Gharehchopogh, S. Mirjalili, “African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems”, Computers & Industrial Engineering, Vol. 158, id. 107408, Aug. 2021.

[46] L.M.K. Al-Ebbini, “An efficient allocation for lung transplantation using ant colony optimization”, Intelligent Automation and Soft Computing, Vol. 35, No. 2, pp. 1971-1985, Nov. 2021.

[47] Y. Wang, T. Li, X.J. Liu, J. Yao, “An adaptive clonal selection algorithm with multiple differential evolution strategies”, Information Sciences, Vol. 604, pp.142-169, Aug. 2022.

[48] S. Rajmohan, E. Elakkiya, S.R. Sreeja, “Multi-cohort whale optimization with search space tightening for engineering optimization problems”, Neural Computing and Applications, Vol. 35, No. 12, pp. 8967-8986, Apr. 2023.

[49] G. Dhiman, “SSC: A hybrid nature-inspired meta-heuristic optimization algorithm for engineering applications”, Knowledge-Based Systems, Vol. 222, id. 106926, Jun. 2021.

[50] J. Derrac, S. Garcı´a, D. Molina, F. Herrera, “A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms”, Swarm and Evolutionary Computation, Vol. 1, No. 1, pp. 3-18, 2011.