Article

Improved Beluga Whale Optimization for Solving the Simulation Optimization Problems with Stochastic Constraints

Shih-Cheng Horng [1,*] and Shieh-Shing Lin [2]

1 Department of Computer Science & Information Engineering, Chaoyang University of Technology, Taichung 413310, Taiwan

2 Department of Electrical Engineering, St. John's University, New Taipei City 251303, Taiwan; sslin@mail.sju.edu.tw

* Correspondence: schong@cyut.edu.tw; Tel.: +886-4-23323000 (ext. 7801)

Abstract: Simulation optimization problems with stochastic constraints are optimization problems with deterministic cost functions subject to stochastic constraints. Solving the considered problem by traditional optimization approaches is time-consuming if the search space is large. In this work, an approach integration of beluga whale optimization and ordinal optimization is presented to resolve the considered problem in a relatively short time frame. The proposed approach is composed of three levels: emulator, diversification, and intensification. Firstly, the polynomial chaos expansion is treated as an emulator to evaluate a design. Secondly, the improved beluga whale optimization is proposed to seek N candidates from the whole search space. Eventually, the advanced optimal computational effort allocation is adopted to determine a superior design from the N candidates. The proposed approach is utilized to seek the optimal number of service providers for minimizing staffing costs while delivering a specific level of care in emergency department healthcare. A practical example of an emergency department with six cases is used to verify the proposed approach. The CPU time consumes less than one minute for six cases, which demonstrates that the proposed approach can meet the requirement of real-time application. In addition, the proposed approach is compared to five heuristic methods. Empirical tests indicate the efficiency and robustness of the proposed approach.

Keywords: beluga whale optimization; ordinal optimization; polynomial chaos expansion; optimal computational effort allocation; emergency department healthcare; average waiting time

MSC: 90-10

Citation: Horng, S.-C.; Lin, S.-S. Improved Beluga Whale Optimization for Solving the Simulation Optimization Problems with Stochastic Constraints. *Mathematics* **2023**, *11*, 1854. https://doi.org/10.3390/math11081854

Academic Editor: Alessandro Niccolai

Received: 21 March 2023
Revised: 11 April 2023
Accepted: 11 April 2023
Published: 13 April 2023

1. Introduction

Simulation optimization problems with stochastic constraints (SOPSC) are optimization problems that optimize a deterministic cost function subject to stochastic constraints on the variables [1]. Such problems become more and more popular over the years in many practical applications, such as dynamic production/inventory lot-sizing problems, power transmission grid reliability problems, signaling-regulatory pathway inference problems, and staffing optimization of emergency department healthcare. The SOPSC belong to the class NP-hard, and their suboptimal designs are usually difficult to solve in a reasonable time [2,3].

The SOPSC are difficult to solve because of the three challenges, (i) a large search space, (ii) ensuring that all constraints are met, and (iii) accurately estimating a stochastic constraint is time-consuming. The ordinal optimization (OO) theory [4,5] has emerged as an efficient technique to handle issues (i) to (iii) simultaneously. Instead of insisting on picking the best design, OO theory focuses on seeking good enough designs and decreases the simulation time significantly. OO theory can seek good enough solutions with high probability through relatively short simulations. The OO theory attempts to resolve the difficulties by employing the following two ideas. (i) The order of a design is more resilient against noise than the value of a design. (ii) Since seeking the best design is computationally expensive, it would be wiser to concentrate on good enough designs. The OO theory has been applied successfully to many situations, including routing optimization in queueing networks [6], staff optimization in multi-skill call centers [7], job-shop scheduling [8], and optimization of shortcuts in the sorting conveyor system [9].

Although the OO theory can expedite the search process by quickly narrowing down the search space, the stochastic constraints still significantly influence the computing efficiency. To decrease the computational time of SOPSC, an algorithm that integrates beluga whale optimization and ordinal optimization (BWOO) is presented to seek a superior design within a short time frame. The BWOO comprises three phases: emulator, diversification, and intensification. First of all, the polynomial chaos expansion (PCE) [10,11] is treated as an emulator to evaluate a design. Secondly, an improved beluga whale optimization (IBWO) is proposed to seek N candidates from the whole search space. Then, an advanced optimal computational effort allocation (AOCBA) is adopted to determine a superior design from the N candidates. These three phases significantly decrease the computing time to solve the SOPSC.

Instead of handling an approximated mathematical model, the optimal staffing cost in emergency department healthcare can be modeled as a SOPSC. Next, the BWOO is utilized to seek the number of staff for minimizing staffing costs while delivering a specific level of care. The target of this SOPSC is to determine the optimal number of service providers to minimize staffing costs while delivering a specific level of care. The contribution of the paper is twofold. First, we develop a BWOO algorithm to seek a superior design of a SOPSC which is short of structural information in a relatively short period. Second, the BWOO algorithm is utilized to determine the optimal staffing cost in emergency department healthcare. The application of the BWOO algorithm is not confined to the SOPSC. The proposed approach can also be utilized to solve computationally expensive simulation optimization problems, discrete probabilistic bicriteria optimization problems, probabilistic constrained simulation optimization problems, and combinatorial stochastic simulation optimization problems.

The remainder of the paper develops as follows. Section 2 introduces the related works of SOPSC. Section 3 illustrates the BWOO algorithm to seek a superior design of a SOPSC. Section 4 introduces the optimal staffing cost in emergency department healthcare, which can be modeled as a SOPSC. Then, the BWOO algorithm is utilized to solve this SOPSC. Section 5 is the comparative analysis of the experimental verification. Finally, the conclusion and further research are presented in Section 6.

2. Literature Review

Popular approaches which are frequently employed to solve SOPSC include the sample path approach, stochastic approximation, and sample average approximation. The sample path approach approximates the output through the average sample observations using a common sequence of random numbers. The stochastic approximation approach approximates the output in an environment where the output is unknown and direct observations are corrupted by noise [12]. The sample average approximation approach uses an approximation scheme by sample averages and replication over several iterations [13]. However, slow convergence rate and trapping in local minima are two drawbacks of the three approaches.

Heuristic algorithms are existing techniques used to solve SOPSC, including tabu search (TS) [14], simulated annealing (SA) [15], genetic algorithm (GA) [16], particle swarm optimization (PSO) [17], differential evolution (DE) [18], biogeography-based optimization (BBO) [19], and social network optimization (SNO) [20]. However, heuristics methods lack the power and flexibility to create ongoing optimal designs. Swarm intelligence (SI) algorithms have fast developed in recent years and applied to solve SOPSC [21]. SI algorithms are inspired by swarms that frequently occur in the real world such as bird flocks, fish schools, and the colony of social insects. The recent novel SI algorithms include golden jackal optimization (GJO) [22], starling murmuration optimizer (SMO) [23], white shark optimizer (WSO) [24], dandelion optimizer (DO) [25], search in forest optimizer (SIFO) [26], snake optimizer (SO) [27], and beluga whale optimization (BWO) [28]. SI algorithms are proven to perform better than conventional optimization approaches and are widely applied in many fields.

BWO is a revolutionary nature-inspired scheme that simulates the attacking and feeding behaviors of beluga whales in nature [28]. The BWO has obvious advantages such as better stability, stronger search ability, higher convergence accuracy, and faster convergence speed. However, BWO has a lack of diversity, which could lead to being trapped in local optimum and premature convergence. To overcome this drawback, the proposed IBWO is developed to accelerate the search process, improve the learning approach, and increase the variety and strengthen the consistency of the chosen candidates.

The optimal staffing costs in the emergency department healthcare can be formulated as a SOPSC. Over the years, common solution approaches for solving the optimal number of staff include the greedy approach, exhaustive search, branch, and bound method, approximate dynamic programming method, and heuristic algorithms [29]. The greedy approach employs the problem-solving heuristic of selecting the design that is optimal locally at each stage in the pursuit of the global optimum [30]. An exhaustive search is simply a brute-force approach to the considered problem. The branch and bound scheme partitions the feasible design space into smaller subsets of designs. However, it is necessary to search all the design spaces if the worst-case occurs. The approximate dynamic programming technique has a complicated design process, which results in a long computing time [31]. Recently, heuristic algorithms are faster and provide near-optimal designs. However, heuristic algorithms still get stuck in being premature and always fall into the local optimum. An intelligent inventory model is proposed to find the optimal service strategy based on the variable conditions along with the optimal quantity and reorder level of the inventory policy [32]. Table 1 shows the research gaps and contributions of the previous author(s).

Table 1. Research gaps and contributions of the previous author(s).

Authors	**Method**	**Category**	**Objectives**
Geiersbach et al. [12]	Stochastic approximation	Gradient-based	Constrained optimization
Zhou et al. [13]	Sample average approximation	Gradient-based	Constrained optimization
Yu et al. [14]	Tabu search	Human	Combinatorial optimization
Cheng [15]	Simulated annealing	Physics	Numerical optimization
Zhang et al. [16]	Genetic algorithm	Evolutionary	Global optimization
Xu et al. [17]	Particle swarm optimization	Swarm	Global optimization
Wang et al. [18]	Differential evolution	Evolutionary	Global optimization
Daneshyar & Charkari [19]	Biogeography-based optimization	Swarm	Global optimization
Beccaria et al. [20]	Social network optimization	Human	Combinatorial optimization
Chopraa & Ansarib [22]	Golden jackal optimization	Swarm	Global optimization
Zamani et al. [23]	Starling murmuration optimizer	Swarm	Global optimization
Braik et al. [24]	White shark optimizer	Swarm	Global optimization
Zhao et al. [25]	Dandelion optimizer	Swarm	Global optimization
Ahwazian et al. [26]	Search in forest optimizer	Swarm	Global optimization
Hashim & Hussien [27]	Snake optimizer	Swarm	Global optimization
Zhong et al. [28]	Beluga whale optimization	Swarm	Global optimization
Sasanfar et al. [29]	Exhaustive search	Other	Combinatorial optimization
Wang et al. [30]	Greedy approach	Other	Combinatorial optimization
Meng et al. [31]	Approximate dynamic programming	Gradient-based	Combinatorial optimization

3. Integrating Beluga Whale Optimization and Ordinal Optimization

3.1. Mathematical Formulation

The SOPSC has the following two challenges, (i) the search space often lacks structural information to identify the optimal design, and (ii) because of the randomness of the constraints, the feasibility of a design cannot certainly be known. The SOPSC are typically shown below.

$$\min h(\mathbf{x}) \tag{1}$$

$$\text{subject to } E[g_i(\mathbf{x})] \le d_i, i = 1, \ldots, I \tag{2}$$

$$\mathbf{Y} \le \mathbf{x} \le \mathbf{U} \tag{3}$$

where $\mathbf{x} = [x_1, \ldots, x_J]^T$ depicts a design vector, $h(\mathbf{x})$ denotes the deterministic cost function, $E[g_i(\mathbf{x})]$ depicts the expectation of the ith constrained function, d_i depicts pre-specified requirement values, I depicts the number of constraints, $\mathbf{Y} = [Y_1, \ldots, Y_J]^T$ and $\mathbf{U} = [U_1, \ldots, U_J]^T$ denote the lower and upper bounds, respectively.

Sufficient replications must be executed to achieve an exact evaluation of $E[g_i(\mathbf{x})]$. However, executing an infinitely long simulation is impossible. Therefore, the following sample mean is an alternative formula to estimate $E[g_i(\mathbf{x})]$.

$$\overline{g}_i(\mathbf{x}) = \frac{1}{L}\sum_{\ell=1}^{L} g_i^{\ell}(\mathbf{x}), i = 1, \ldots, I \tag{4}$$

where L depicts the amount of replications, and $g_i^{\ell}(\mathbf{x})$ represents the estimation of the ℓth replication. When the number of replications increases, the sample mean $\overline{g}_i(\mathbf{x})$ will have a preferable estimation of $E[g_i(\mathbf{x})]$. That is, a larger value of L is more closely approximate $E[g_i(\mathbf{x})]$.

Since the constraints are usually soft ones, the SOPSC is imposed by adding an extra penalty [33]. An infeasible design is penalized so that its chance of survival is much decreased as compared to a feasible design.

$$\min f(\mathbf{x}) = h(x) + \eta \times \sum_{i=1}^{I} pe_i(\mathbf{x}) \tag{5}$$

where η depicts a penalty factor, $f(\mathbf{x})$ depicts a penalized cost function, and $pe_i(\mathbf{x})$ represents the quadratic penalty function.

$$pe_i(\mathbf{x}) = \begin{cases} 0, & if \overline{g}_i(\mathbf{x}) \le d_i, \\ (\overline{g}_i(\mathbf{x}) - d_i)^2, & \text{else}, \end{cases} i = 1, \ldots, I. \tag{6}$$

The penalty factor is usually a positive constant that is large enough to enlarge the penalty function whenever a constraint is violated. Let L_a represent the sufficient large value of L, and the exact evaluation of (4) is calculated using $L = L_a$. For simplicity, we let $f_a(\mathbf{x})$ indicate the penalized cost function of $\mathbf{x}$ through exact evaluation.

The OO theory [4] elaborates that orders of designs are still retained even though they are evaluated by a crude model. Therefore, the PCE emulator is utilized to estimate a design more quickly. Thus, the IBWO cooperated with the PCE emulator is employed to look for N candidates from the whole search space.

3.2. Polynomial Chaos Expansion

The emulator is an important and growing field of research that signifies a major achievement in surrogate modeling, including the support vector regression [34], multivariate adaptive regression splines [35], extreme learning machines [36], regularized minimal-energy tensor-product splines [37], and polynomial chaos expansions (PCE) [10,11]. Among them, PCE builds a polynomial approximation of a model whose inputs are random variables. There are three advantages of PCE: (i) it allows for uncertainty quantification of input parameters, (ii) it can be evaluated much faster than the stochastic response itself, and (iii) its exact analytical expression. PCE has been widely adopted in various applications, including curve fitting, forecasting, prediction, and function approximation [10]. Therefore, the PCE emulator is used to quickly evaluate a design. The PCE with second-order chaos polynomial factor is composed of three layers as shown in Figure 1.

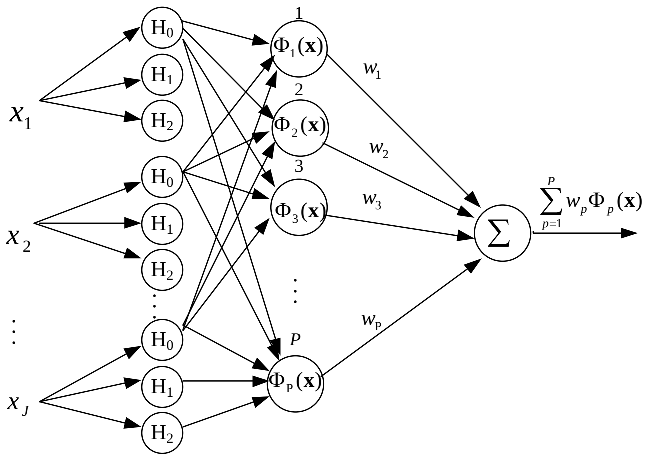


Figure 1. Framework of a PCE with second-order chaos polynomial factor.

The PCE utilizes orthogonal polynomials as a basis for the fitting of response outputs based on a probabilistic data set. We randomly sample Π $\mathbf{x}$'s from search space and evaluate $\boldsymbol{F_a}(\hat{\mathbf{x}})$ using exact evaluation, where $\hat{\mathbf{x}} = \frac{\mathbf{x}-\mu}{\sigma}$ are the normal standard of $\mathbf{x}$, and μ and σ represent the mean and standard deviation, respectively. We represent these Π sampled designs as $(\hat{\mathbf{x}}_i,\ F_a(\hat{\mathbf{x}}_i))$. The PCE can approximate $F(\hat{\mathbf{x}})$ using sums of orthonormal polynomials.

$$F(\hat{\mathbf{x}}) = \sum_{p=1}^{P} w_p \Phi_p(\hat{\mathbf{x}}) \tag{7}$$

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

$$\Phi_p(\hat{x}) = \prod_{k=1}^{K} H_p(\hat{x}_k) \tag{8}$$

where K is the dimension of a multivariate orthogonal polynomial, which is obtained by the input data through the Hermite polynomials $H_p(\cdot)$. These data points of $\Phi_p(\hat{x})$ can be extracted from the input variables in the modeling process through the Hermite polynomials. 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-square-minimization method is used to determine the expansion coefficients w_p, $p = 1, \ldots, P$.

$$\begin{bmatrix} w_1 \\ \vdots \\ w_P \end{bmatrix} = \left[\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right]^{-1}\boldsymbol{\Phi}^{\mathrm{T}} \begin{bmatrix} F_a(\hat{\mathbf{x}}_1) \\ \vdots \\ F_a(\hat{\mathbf{x}}_\Pi) \end{bmatrix} \tag{9}$$

The setting of Π must be larger than the setting of P, i.e., $\Pi > P$. The matrix $\boldsymbol{\Phi}$ is determined as follows.

$$\boldsymbol{\Phi} = \begin{bmatrix} \Phi_1(\hat{\mathbf{x}}_1) & \Phi_2(\hat{\mathbf{x}}_1) & \cdots & \Phi_P(\hat{\mathbf{x}}_1) \\ \vdots & \vdots & & \vdots \\ \Phi_1(\hat{\mathbf{x}}_\Pi) & \Phi_2(\hat{\mathbf{x}}_\Pi) & \cdots & \Phi_P(\hat{\mathbf{x}}_\Pi) \end{bmatrix} \tag{10}$$

The PCE is trained offline to significantly decrease the computing burden. After training the PCE, the model can be generalized with a new design $\mathbf{x}$ to predict $F(\hat{\mathbf{x}})$.

3.3. Improved Beluga Whale Optimization

In the diversification phase, we can adapt state-of-the-art optimization techniques with the assistance of the PCE to look for N candidates from the whole search space. Since the BWO explores several regions at the same time, it is more suitable for the specific requirements. In essence, BWO uses the following three behaviors: pair swim, prey, and whale fall. The pair swims behavior is corresponding to exploration. Beluga whales engage in social interactions under different postures, such as two beluga whales swimming in close pairs in a synchronized or mirrored manner. The preying behavior is corresponding to exploitation. Beluga whales cooperatively feed and move based on the location of nearby companions. Beluga whales prey by sharing each other's location information, considering the top candidates and others. Exploration is related to global search as well as exploitation is related to local search. In the first one, we are interested in exploring the search space looking for good solutions, whereas, in the second one, we want to refine the solution and try to avoid big jumps in the search space. The whale fall is corresponding to imitating small changes in the groups. During the migration and foraging, some beluga whales do not survive and fall into the depths of the ocean.

The proposed IBWO has three algorithmic parameters, including a balance factor between exploration and exploitation (B_f), the probability of whale fall (W_f), and the jump strength of Levy flight (C_f). BWO has a lack of diversity, which could lead to being trapped in local optimum and premature convergence. To overcome these drawbacks, the three algorithmic parameters B_f, $W_{f,}$ and C_f are iteratively modified to intensify exploration in the former process and exploitation in the latter process. The variation of B_f decreases exponentially with increased iterations. A large B_f focuses on finding promising regions at the beginning, then a small B_f focuses on searching near already found promising designs near the end. The variation of W_f and C_f are also exponentially decreased as iterations increase to strengthen exploitation.

The following notations are used in IBWO. Ψ depicts the number of beluga whales, $t_{\max}$ denotes the maximum number of iterations, $\mathbf{x}_i^t = [x_{i,1}^t, \ldots, x_{i,J}^t]^{\mathrm{T}}$ and $\mathbf{r}_i^t = [r_{i,1}^t, \ldots, r_{i,J}^t]^{\mathrm{T}}$ are the positions of the ith beluga whale and a randomly selected beluga whale at iteration t, respectively, and $\mathbf{x}^* = [x_1^*, \ldots, x_J^*]^{\mathrm{T}}$ is the position of the elite beluga whale. ${B_f}^t \in [B_{f\_\min}, B_{f\_\max}]$, ${W_f}^t \in [W_{f\_\min}, W_{f\_\max}]$, and ${C_f}^t \in [C_{f\_\min}, C_{f\_\max}]$ depict the balance factor B_f, probability W_f, and jump strength C_f at iteration t, respectively, where $B_{f\_\min}$, $W_{f\_\min}$, $C_{f\_\min}$ are lower bound, and $B_{f\_\max}$, $W_{f\_\max}$, $C_{f\_\max}$ are upper bound.

The details of the IBWO algorithm are explained as follows (Algorithm 1).

Algorithm 1: The IBWO

Step 1: Configuration of parameters
Set parameters to Ψ, $B_{f\_\min}$, $B_{f\_\max}$, $W_{f\_\min}$, $W_{f\_\max}$, $C_{f\_\min}$, $C_{f\_\max}$, and $t_{\max}$. Create an index variable t and initialize it to 0.
Step 2: Initialize the population
Initialization of a population with Ψ beluga whales.

$$\mathbf{x}_i^0 = \mathbf{Y} + \lfloor rand[0,1] \times (\mathbf{U} - \mathbf{Y}) \rfloor, i = 1, \ldots, \Psi. \tag{11}$$

where $rand[0,1]$ is a random number in the range 0 to 1, and $\mathbf{Y}$ and $\mathbf{U}$ represent the lower and upper bounds, respectively.
Step 3: Ranking

(a) Compute $f(\mathbf{x}_i^t)$ of every beluga whale cooperated with PCE, $i = 1, \ldots, \Psi$.
(b) Sort the Ψ beluga whales on the basis of their fitness from the least to the biggest, then determine the elite $\mathbf{x}^*$.

Step 4: Modify three algorithmic parameters

$$\mathrm{B}_f{}^t = \mathrm{B}_{f\_\min} + \left(\mathrm{B}_{f\_\max} - \mathrm{B}_{f\_\min}\right) \times \exp\left(\ln\left(\frac{\mathrm{B}_{f\_\min}}{\mathrm{B}_{f\_\max}}\right) \times \frac{t}{t_{\max}}\right) \tag{12}$$

Algorithm 1: Cont.

$$\mathrm{W}_f{}^t = \mathrm{W}_{f\_max} \times \exp\left(-\sqrt{\frac{\mathrm{W}_{f\_max}}{\mathrm{W}_{f\_min}}} \times \frac{t}{t_{max}}\right) \tag{13}$$

$$C_f^t = C_{f\_\min} + (C_{f\_\max} - C_{f\_\min}) \times \left(1 - \exp\left(\frac{C_{f\_\max}}{C_{f\_\min}} \times \left(\frac{t}{t_{\max}} - 1\right)\right)\right) \tag{14}$$

Step 5: Exploration and exploitation
If $\mathrm{B}_f{}^t$ > 0.5, perform exploration.

$$x_{i,j}^{t+1} = \begin{cases} x_{i,p}^t + (x_{r,q}^t - x_{i,p}^t) \times (1 + rand[0,1]) \times \sin(2\pi \cdot rand[0,1]), j = even \\ x_{i,p}^t + (x_{r,q}^t - x_{i,p}^t) \times (1 + rand[0,1]) \times \cos(2\pi \cdot rand[0,1]), j = odd \end{cases}, i = 1, \ldots, \Psi \tag{15}$$

where r is an arbitrarily chosen beluga whale, p and q are random numbers selected from J-dimension, when $x_{i,j}^{t+1} < Y_j$, set $x_{i,j}^{t+1} = Y_j$, and when $x_{i,j}^{t+1} > U_j$, set $x_{i,j}^{t+1} = U_j$.
Else if $\mathrm{B}_f{}^t \leq 0.5$, perform exploitation.

$$\mathbf{x}_i^{t+1} = rand[0,1] \cdot \mathbf{x}^* - rand[0,1] \cdot \mathbf{x}_i^t + C^t{}_f \cdot L_F \cdot (\mathbf{x}_r^t - \mathbf{x}_i^t), i = 1, \ldots, \Psi. \tag{16}$$

where r is an arbitrarily chosen beluga whale, $\mathbf{x}^*$ is the elite beluga whale, $C^t{}_f$ denotes the jump strength of Levy flight, and L_F denotes the following Levy flight function,

$$\mathrm{L_F}= 0.05 \times \frac{\mathrm{u}}{|\mathrm{v}|^{\frac{1}{\beta}}} \times \left(\frac{\Gamma(1+\beta) \times \sin(\pi\beta/2)}{\Gamma((1+\beta)/2) \times \beta \times 2^{\frac{\beta-1}{2}}}\right)^{\frac{1}{\beta}} \tag{17}$$

where u and v indicate the mean and standard derivation in Gauss distribution with u = 0, Γ denotes the Gamma function, and β = 1.5 is a default value. When $x_{i,j}^{t+1} < Y_j$, set $x_{i,j}^{t+1} = Y_j$, and when $x_{i,j}^{t+1} > U_j$, set $x_{i,j}^{t+1} = U_j$.
Step 6: Whale fall
If $\mathrm{B}_f{}^t \leq \mathrm{W}_f{}^t$,

$$\mathbf{x}_i^{t+1} = rand[0,1] \cdot \mathbf{x}_i^t - rand[0,1] \cdot \mathbf{x}_r^t + rand[0,1] \cdot (\mathbf{U} - \mathbf{V}) \cdot \mathrm{e}^{-2\Psi W_f^t \times \frac{t}{t_{\max}}}, i = 1, \ldots, \Psi. \tag{18}$$

where r is a randomly selected beluga whale. When $x_{i,j}^{t+1} < Y_j$, set $x_{i,j}^{t+1} = Y_j$, and when $x_{i,j}^{t+1} > U_j$, set $x_{i,j}^{t+1} = U_j$.
Step 7: Replace elitism
Compute $F(\mathbf{x}_i^{t+1})$ and $F(\mathbf{x}^*)$ cooperated with PCE and adopt the greedy approach between $\mathbf{x}_i^{t+1}$ and $\mathbf{x}^*$. If $F(\mathbf{x}_i^{t+1}) < F(\mathbf{x}^*)$, set $\mathbf{x}^* = \mathbf{x}_i^{t+1}$.
Step 8: Termination
If $t \geq t_{\max}$, terminate; else, set $t = t + 1$ and return to Step 2.

The IBWO stops after the $t_{\max}$ iterations have been executed. When the IBWO is suspended, the Ψ beluga whales are ordered on the basis of their fitness. Although the IBWO is designed for *continuous variable*s, a real value can be rounded to the nearest integer through the bracket function $z_{i,j}^{t_{\max}} = \left\lfloor x_{i,j}^{t_{\max}} \right\rfloor$, where $x_{i,j}^{t_{\max}} \in \Re$ and $z_{i,j}^{t_{\max}} \in Z$. Then, the prior N beluga whales are chosen to constitute the candidate subset.

3.4. Advanced Optimal Computing Budget Allocation

To increase the computing efficiency of the original OCBA, the AOCBA is utilized to determine a superior design from the candidate subset. The AOCBA allocates the computational effort sequentially to all the competing alternatives based on the means and variances. In the original OCBA, all replications must be performed at every iteration to calculate the statistics of competing alternatives. The AOCBA just needs to carry out incremental replications every iteration to calculate the statistics of competing alternatives. In the AOCBA, more computing budget is allocated to simulating critical alternatives, and less is allocated to non-critical alternatives. Emphasizing little critical alternatives not only saves computational effort but also reduces the variances of critical alternatives. AOCBA is developed to improve the computing efficiency of OO by distributing the computational effort reasonably. OO theory allocates identical computational effort to every competing alternative, while AOCBA allocates computational effort to a competing alternative based on its performance. Thus, AOCBA proposes a way of asymptotically optimal allocation of computing budget among competing alternatives.

The number of incremental replications can be allocated to critical designs through the statistics obtained from the N candidates. Let C_a represent the available computational effort, L_0 indicate the essential replications allocated to each candidate, and L_n denote the replications assigned to the nth candidate. A one-time incremental computational effort, Δ, is provided in every iteration. Typically, the best setting of Δ is problem-dependent and can be found by experimentation. A large setting of Δ results in a waste of computational effort to accomplish an unnecessarily high confidence level, while a small setting of Δ performs the allocating procedure many times. The AOCBA aims at maximizing the probability of correct selection given that $L_1 + L_2 + \cdots + L_N = C_a$ by intelligently allocating C_a to $L_1, \ldots, L_N$. The available computational budget C_a is defined as $C_a = \frac{N \times L_a}{\tau}$, where L_a is the replications adopted in the exact evaluation, and τ depicts a speed-up factor [38,39] (Algorithm 2).

Algorithm 2: The AOCBA

Step 1. Define the values of L_0, set $l = 0$, $L_n^l = L_0$, $n = 1, \ldots, N$, and calculate the available computational effort $C_a = \frac{N \times L_a}{\tau}$.

Step 2. Add a one-time incremental computing budget Δ to $\sum_{n=1}^{N} L_n^l$, and update the replications.

$$L_j^{l+1} = (\sum_{n=1}^{N} L_n^l + \Delta) \times \theta_j^l / (\theta_b^l + \sum_{n=1, n \neq b}^{N} \theta_n^l) \tag{19}$$

$$L_b^{l+1} = \frac{\theta_b^l}{\theta_j^l} \times L_j^{l+1} \tag{20}$$

$$L_n^{l+1} = \frac{\theta_n^l}{\theta_j^l} \times L_j^{l+1} \tag{21}$$

where $\frac{\theta_n^l}{\theta_j^l} = \left(\frac{\delta_n^l \times (\overline{f}_b^l - \overline{f}_j^l)}{\delta_j^l \times (\overline{f}_b^l - \overline{f}_n^l)} \right)^2$, $\theta_b^l = \delta_b^l \sqrt{\sum_{n=1, n \neq b}^{N} \left(\frac{\theta_n^l}{\delta_n^l} \right)^2}$, $\overline{f}_n^l = \frac{1}{L_n^l} \sum_{k=1}^{L_n^l} f_k(\mathbf{x}_n)$,
$\delta_n^l = \sqrt{\frac{1}{L_n^l} \sum_{h=1}^{L_n^l} \left(f_k(\mathbf{x}_n) - \overline{f}_n^l \right)^2}$ for all $n \neq j \neq b$,$b = \arg\min_n \overline{f}_n^l$, $\mathbf{x}_n$ represents the nth candidate, and $f_k(\mathbf{x}_n)$ denotes the penalized objective value of $\mathbf{x}_n$ at the kth replication.

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

$$\hat{f}_n^{l+1} = \frac{1}{(L_n^{l+1} - L_n^l)} \sum_{k=L_n^l+1}^{L_n^{l+1}} f_k(\mathbf{x}_n) \tag{22}$$

$$\hat{\delta}_n^{l+1} = \sqrt{\frac{1}{(L_n^{l+1} - L_n^l)} \sum_{k=L_n^l+1}^{L_n^{l+1}} \left(f_k(\mathbf{x}_n) - \hat{f}_n^{l+1} \right)^2} \tag{23}$$

Step 4. Compute the updated mean ($\overline{f}_n^{l+1}$) and updated standard deviation (δ_n^{l+1}) of the nth candidate for overall replications.

$$\overline{f}_n^{l+1} = \frac{1}{L_n^{l+1}} \left(L_n^l \times \overline{f}_n^l + (L_n^{l+1} - L_n^l) \times \hat{f}_n^{l+1} \right) \tag{24}$$

$$\delta_n^{l+1} = \sqrt{\frac{1}{(L_n^{l+1} - 1)} \times \left(L_n^l \left(\overline{f}_n^l \right)^2 + (L_n^l - 1)(\delta_n^l)^2 + (L_n^{l+1} - L_n^l) \left(\hat{f}_n^{l+1} \right)^2 + (L_n^{l+1} - L_n^l - 1) \left(\hat{\delta}_n^{l+1} \right)^2 - L_n^{l+1} \left(\hat{f}_n^{l+1} \right)^2 \right)} \tag{25}$$

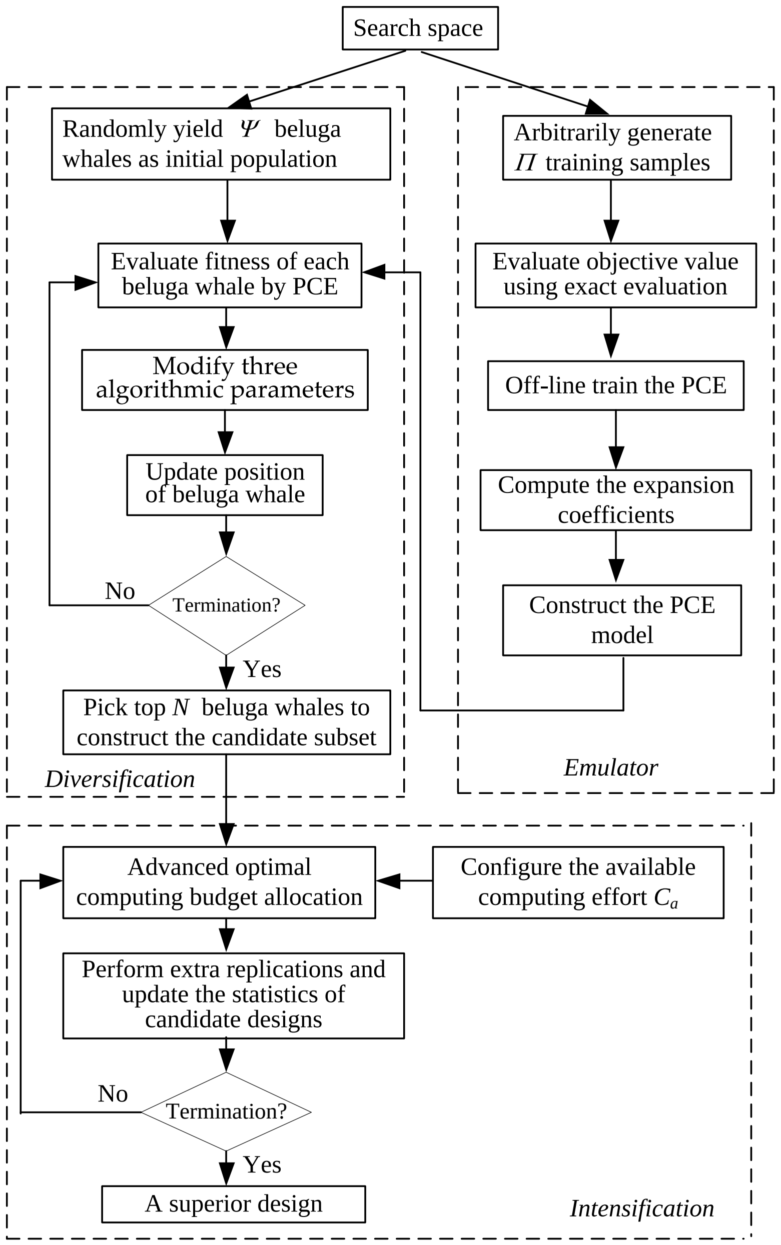
Step 5. If $\sum_{n=1}^{N} L_n^l \geq C_a$, stop and determine the optimal $\mathbf{x}^*$ with the minimum objective value; else, let $l = l + 1$ and go to Step 1.

3.5. The BWOO Algorithm

The flowchart of the BWOO algorithm (Algorithm 3) is presented in Figure 2.

Algorithm 3: The BWOO

Step 1: Define the values of Ψ, $B_{f\_\min}$, $B_{f\_\max}$, $W_{f\_\min}$, $W_{f\_\max}$, $C_{f\_\min}$, $C_{f\_\max}$, $t_{\max}$, N, L_a, L_0, and Δ.
Step 2: Randomly select Π $\mathbf{x}$'s from the search space, evaluate $f_a(\mathbf{x})$ using exact evaluation, and train the PCE offline using these Π designs.
Step 3: Generate $\Psi\mathbf{x}$'s to be the initial population, then apply the IBWO algorithm to those beluga whales that cooperated with PCE. After the IBWO algorithm terminates, rank all the final $\Psi\mathbf{x}$'s based on their approximate fitness from the lowest to the highest, and choose the prior $N\mathbf{x}$'s to construct a candidate subset.
Step 4: Apply the AOCBA algorithm to the N candidates and determine the optimum $\mathbf{x}^*$, which is the superior design.



Search space
Randomly yield Ψ beluga whales as initial population
Evaluate fitness of each beluga whale by PCE
Modify three algorithmic parameters
Update position of beluga whale
No
Termination?
Yes
Pick top N beluga whales to construct the candidate subset
Diversification
Arbitrarily generate Π training samples
Evaluate objective value using exact evaluation
Off-line train the PCE
Compute the expansion coefficients
Construct the PCE model
Emulator
Advanced optimal computing budget allocation
Configure the available computing effort C_a
Perform extra replications and update the statistics of candidate designs
No
Termination?
Yes
A superior design
Intensification

Figure 2. Flowchart of the BWOO algorithm.

4. Optimal Staffing Cost in the Emergency Department Healthcare

4.1. Emergency Department Healthcare

Most emergency departments have a recognizable patient arrival pattern, which follows a process as depicted in Figure 3. The patient flow process is modeled through a discrete-event simulation modeling with the following five assumptions. (i) The arrival patient to the reception follows a nonstationary Poisson process with a rate of λ(t). (ii) The arrival patient to the examination room follows a Poisson process with a constant rate. (iii) The routing probabilities of various patients are given at each location. (iv) The number of staff at each location decides the allocation of the system. (v) The distribution of service time and the rates are given.

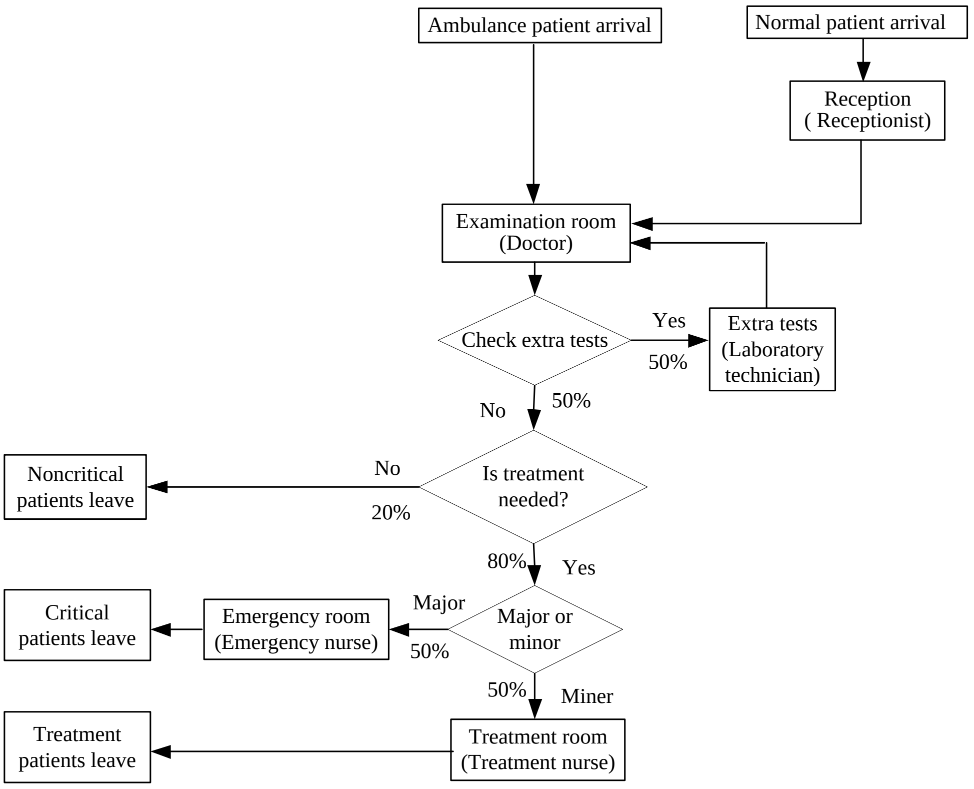

Ambulance patient arrival
Normal patient arrival
Reception (Receptionist)
Examination room (Doctor)
Check extra tests
Yes
50%
Extra tests (Laboratory technician)
No
50%
Noncritical patients leave
No
20%
Is treatment needed?
80%
Yes
Critical patients leave
Emergency room (Emergency nurse)
Major
50%
Major or minor
50%
Miner
Treatment patients leave
Treatment room (Treatment nurse)

Figure 3. Patient flow process of an emergency department.

Now, the optimal staffing cost in the emergency department healthcare is formulated as a SOPSC as follows.

$$\min h(\mathbf{x}) \tag{26}$$

$$\text{subject to } E[g_1(\mathbf{x})] \le d_1 \tag{27}$$

$$E[g_2(\mathbf{x})] \le d_2 \tag{28}$$

$$\mathbf{Y} \le \mathbf{x} \le \mathbf{U} \tag{29}$$

where $\mathbf{x} = [x_1, \ldots, x_5]^T$ indicates a design, x_1~x_5 depicts the number of receptionists, doctors, laboratory technicians, treatment nurses, and emergency nurses, respectively, $E[g_1(\mathbf{x})]$ represents the average waiting time of critical patients, d_1 is prespecified requirement values of critical patients, $E[g_2(\mathbf{x})]$ represents the average waiting time of treatment patients, d_2 is prespecified requirement values of treatment patients, $h(\mathbf{x})$ is the staffing cost, and $\mathbf{Y}$ and $\mathbf{U}$ denote the lower and upper bounds, respectively.

The target of the SOPSC is to find the optimal number of staff $\mathbf{x}^*$ to minimize staffing cost $h(\mathbf{x})$ subject to integrality conditions, two constraints, and limits of staff members. The sample mean is one of the most common alternatives to estimate the value of $E[g_i(\mathbf{x})]$.

$$\overline{g}_i(\mathbf{x}) = \frac{1}{L}\sum_{\ell=1}^{L} g_i^{\ell}(\mathbf{x}), i = 1,2. \tag{30}$$

where L represents the quantity of replications, and $g_i^{\ell}(\mathbf{x})$ is the estimation of the ℓth replication. Since the constraints are soft, the penalty function is employed for handling the two inequality constraints.

$$\min f(\mathbf{x}) = h(x) + \eta \times \sum_{i=1}^{2} pe_i(\mathbf{x}) \tag{31}$$

where η depicts a penalty factor, $f(\mathbf{x})$ is a penalized cost function, and $pe_i(\mathbf{x})$ denotes the quadratic penalty function.

$$pe_i(\mathbf{x}) = \begin{cases} 0, & if\ \overline{g}_i(\mathbf{x}) \le d_i, \\ (\overline{g}_i(\mathbf{x}) - d_i)^2, \text{else}, \end{cases} i = 1,2. \tag{32}$$

Figure 4 describes the input/output relationship of the emergency department healthcare, where $\mathbf{x}$ depicts a design and $f(\mathbf{x})$ depicts the penalized cost function. Let L_a indicate the sufficiently large value of L, and the exact evaluation of (31) is defined as $L = L_a$. For simplicity, $f_a(\mathbf{x})$ is denoted as the penalized cost function of $\mathbf{x}$ obtained by an exact evaluation.

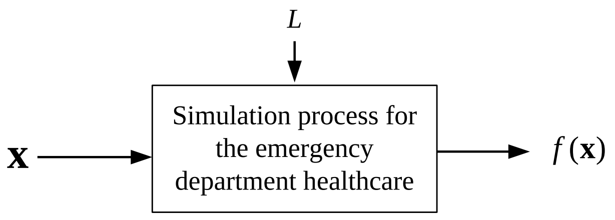


Figure 4. The input/output relationship of the emergency department healthcare.

4.2. *Application of the BWOO Method*

4.2.1. Constitute the Emulator

Four procedures were utilized for constructing the PCE emulator to evaluate a design. (i) Arbitrarily chose Π $\mathbf{x}$'s from search space and calculate $f_a(\mathbf{x})$ by exact evaluation, then indicate these Π designs and their estimations as $\mathbf{x}_i$ and $f_a(\mathbf{x}_i)$, respectively. (ii) Set the number of PCE terms, i.e., P = 2. (iii) Pre-compute the matrix of multivariate orthogonal polynomial basis functions. (iv) Compute the expansion coefficients.

4.2.2. Construct the Candidate Subset

With the assistance of the PCE emulator, N candidates were selected by the IBWO. First of all, Ψ beluga whales were randomly generated to be the initial population. The fitness of a beluga whale was calculated using the PCE emulator. When the IBWO stopped after $t_{\max}$ iterations, the Ψ beluga whales were sorted based on their fitness. The former N beluga whales were selected to constitute the candidate subset.

4.2.3. Find the Superior Design

Eventually, the AOCBA method was adopted to seek a superior design from the N candidates. In general, the number of N cannot be too large to improve efficiency. On the other hand, some outstanding designs will miss when the setting of N is too small. Reference [38] suggested that a suitable value of L_0 is between 5 to 20, and an appreciate the value of Δ is smaller than 100 but larger than 10% of N.

5. Practical Applications

5.1. Practical Example

A practical example of an emergency department adopted and extended from the stochastic resource problem 3 in [40] is used to verify the BWOO method. Because of operating cost considerations, at most 5 receptionists, 7 doctors, 6 laboratory technicians, 8 treatment nurses, and 10 emergency nurses can be employed. The target is to find how many staff members could be employed to minimize staffing costs while delivering a specific level of care. We assume that receptionists, doctors, laboratory technicians, and both treatment nurses and emergency nurses earn $40, $120, $50, and $30, respectively.

The arrival pattern of walk-in patients follows a nonstationary Poisson process based on Table 2. The arrival pattern of ambulance patients follows a Poisson process with a constant rate of 2 per hour. Distributions of service time at each stage are listed in Table 3. The lower and upper bounds are the two parameters in the parentheses of the uniform distribution. The min, mode, and max are the three parameters in the parentheses of the triangular distribution. We run for 100 more days and adopt a four-day warm-up period. We have conducted six cases of different parameters d_1 and d_2. Parameters d_1 and d_2 indicate pre-specified requirements of the average waiting time for critical patients and treatments for patients, respectively. The six cases are obtained by permutations using different values of d_1 (2, 2.5, and 3 h) and d_2 (2 and 2.5 h).

Table 2. Walk-in arrival rates.

t	**0**	**2**	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**	**22**
$\lambda(t)$	5.25	3.8	3	4.8	7	8.25	9	7.75	7.75	8	6.5	3.25

Table 3. Service time distributions.

Location	**Distribution**
Reception	Uni (5,10)
Extra tests	Tri (10,20,30)
Examination	Uni (10,20)
Re-examination	Uni (7,12)
Treatment	Uni (20,30)
Emergency	Uni (60,120)

There are 8549 arbitrarily selected designs to train the PCE. The number of samples Π = 8549 was obtained by the sampling size formula using a confidence interval of 1% and a confidence level of 95% [41]. The performance of a sample was evaluated by an exact evaluation.

The penalty factor was $\eta = 10^4$. A large penalty factor is assured to amplify the penalty function for infeasible designs. The lower and upper bounds were $\mathbf{Y} = [1,1,1,1,1]^T$ and $\mathbf{U} = [5,7,6,8,10]^T$, respectively. Thus, the size of the search space is 77,760. The parameters used in IBWO were $B_{f\_\min} = 0.5$, $B_{f\_\max} = 1$, $W_{f\_\min} = 0.05$, $W_{f\_\max} = 0.3$, $C_{f\_\min} = 0.1$, $C_{f\_\max}= 2$, $t_{\max} = 100$, and $\Psi = 40$. Various hand-tuned experiments demonstrate that IBWO utilizing the above parameters is well-performed. Figure 5 illustrates the curves of three factors B_f, W_f, and C_f over 100 iterations. The IBWO explored the search space in preceding iterations as well as exploited the certain region in later iterations. The number of candidates was N = 10. The parameters used in AOCBA were L_0=20, Δ=10 and $L_a = 10^4$. The speed-up factor τ corresponding to N = 10 is 3.4 [38]. Thus, the available computing effort C_a was 29,412.

Table 4 presents the superior design $\mathbf{x}^*$, cost, and CPU times of six cases. For example, the optimization model in Case IV gives a design that costs $630 and staff assignment as follows: 1 receptionist, 3 doctors, 1 laboratory technician, 3 treatment nurses, and 3 emergency nurses, such that a limited average waiting time for both critical patients and treatment of patients of 2.5 h. Figure 6 displays the convergence curve of the best-so-far candidate solution for Case IV. The CPU time consumes less than one minute for six cases, which demonstrates that the BWOO can meet the requirement of real-time application.

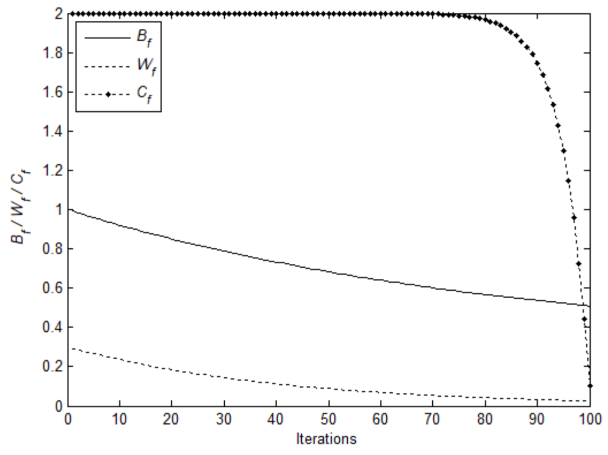


Figure 5. Variations of B_f, W_f, and C_f over iterations.

Table 4. The superior design $\mathbf{x}^*$, cost, and CPU times of six cases.

Case	d_1	d_2	$\mathbf{x}^*$	Cost	CPU Time (s)
I	2	2	$[3,4,4,3,8]^T$	1130	57.3
II	2	2.5	$[2,5,1,4,8]^T$	1090	55.8
III	2.5	2	$[2,3,2,3,6]^T$	810	56.9
IV	2.5	2.5	$[1,3,1,3,3]^T$	630	57.2
V	3	2	$[1,3,4,3,6]^T$	870	55.5
VI	3	2.5	$[1,3,1,2,2]^T$	570	56.4

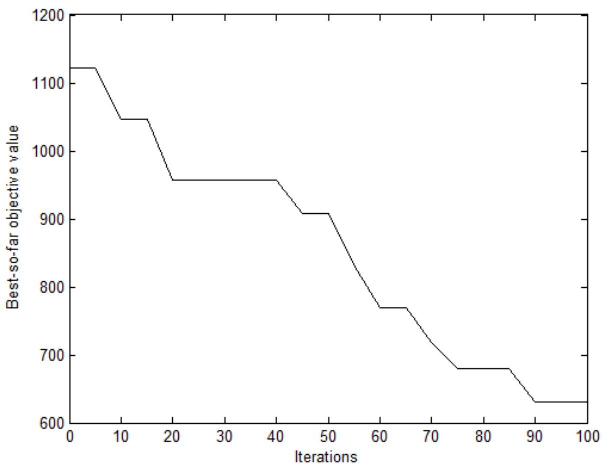


Figure 6. The convergence curve of the best-so-far candidate solution for Case IV.

5.2. Performance Comparison

The BWOO algorithm was compared to five metaheuristic methods for case I: GA [14], ant colony optimization (ACO) [42], clonal selection algorithm (CSA) [43], whale optimization algorithm (WOA) [44], and equilibrium optimization (EO) [45]. A population size of 40, roulette wheel selection, single-point crossover with a crossover probability of 0.8, and uniform mutation with a mutation rate of 0.02 were adopted in the GA. In the employed ACO, a population size of 40, an initial pheromone of 0.1, a global pheromone volatile factor of 0.3, the local pheromone evaporation rate of 0.5, the relative importance of information with 1, and the control factor between the relative proportion of the exploitation and biased exploration with 0.9 were utilized. A population size of 40, a strength of mutation of 10, and a receptor editing rate of 0.05 were adopted in the CSA. In the WOA, a population size of 40, and the shape of a logarithmic spiral of 2 were employed. In the EO, a population of particles of 40, a generation rate of 0.5, a diversification factor of 3, and the exploitation factor of 1 were employed.

The exact evaluation was used to calculate the objective value for five metaheuristic methods. Because of randomness, 30 trials were conducted to verify the reliability of six methods. Since the five metaheuristic methods need more computation times to seek the optimum, the search processes terminated after they had spent 30 min of computation time. Table 5 illustrates the statistical results and average CPU times over 30 trials for 6 approaches. The averages of the best-so-far objective value obtained by GA, ACO, CSA, WOA, and EO were 13.16%, 15.79%, 18.42%, 10.53%, and 11.40% larger than that obtained by BWOO, respectively. Experimental results illustrate that the BWOO outperforms five metaheuristic methods.

Table 5. Statistic results and average CPU times of six methods.

Methods	Min.	Max.	AOV [†]	$\frac{\text{AOV}^{\dagger}-*^{\S}}{*}\times 100\%$	S.D.	S.E.M.	Average Rank Percentage	Average CPU Time (s)
BWOO	1120	1160	1140	0	15	2.73	0.02%	57.6
GA with exact evaluation	1240	1350	1290	13.16%	50	9.13	4.84%	1797
ACO with exact evaluation	1230	1390	1320	15.79%	85	15.52	5.72%	1800
CSA with exact evaluation	1310	1430	1350	18.42%	70	12.78	7.37%	1795
WOA with exact evaluation	1220	1310	1260	10.53%	40	7.30	2.21%	1798
EO with exact evaluation	1240	1340	1270	11.40%	45	8.22	3.95%	1799

[†] AOV: average of the best-so-far objective value; [§] $*$: AOV obtained by BWOO.

Finally, an analysis concerning rank percentage was conducted to illustrate the rank of a superior design in the search space. Because it is impossible to decide the ranks of all designs, a representative subset, Ω, is constructed to represent the characteristics of the large search space. The rank percentage of a superior design is defined as $\frac{r}{|\Omega|} \times 100\%$, where r indicates the rank of a superior design in Ω. Generally, 11,556 designs were arbitrarily chosen from the whole search space to constitute the representative subset. The objective values of all samples were computed using exact evaluation. The value of $|\Omega|$ = 11,556 was calculated by the sampling size formula with a confidence interval of 1% and a confidence level of 98% [41]. Table 5 also illustrates the average rank percentages resulting from 6 methods. The standard error of the mean (S.E.M.) resulting from BWOO was 2.73. The small S.E.M. illustrates that most of the superior designs resulting from the BWOO are fairly close to the optimum over 30 trials.

6. Conclusions and Outlooks

To solve the SOPSC in a reasonable time, an algorithm integrating BWO into OO was developed. The BWOO composes of three phases: emulator, diversification, and intensification. The PCE emulator was efficient in rapidly evaluating a design. The BWOO adopted the IBWO for diversification and the AOCBA for intensification. The BWOO was adopted for the optimal staffing cost in the emergency department healthcare, which is modeled as a SOPSC. A practical emergency department with six cases was utilized to test the BWOO algorithm. The CPU time consumes less than one minute for six cases, which demonstrates that the BWOO can meet the real-time requirement. The BWOO was compared to five metaheuristic methods—GA, ACO, CSA, WOA, and EO cooperated with an exact evaluation. Test results demonstrated that most of the superior designs resulting from the BWOO are fairly close to the optimum over 30 trials. Since the BWOO usually obtains a near optimum in a reasonable time, the limitation of the proposed method is that it does not provide a globally optimal solution. The PCE emulator can be replaced by the essential replications L_0 adopted in the AOCBA to resolve this limitation. Futures researches will focus on applying OO to resolve stochastic dominance-constrained optimization problems, such as risk-averse stochastic optimization problems and conditional value-at-risk optimization problems.

Author Contributions: S.-C.H. Horng designed and conceived the experiments; S.-C.H. Horng performed the experiments; S.-S.L. Lin analyzed the data; S.-S.L. Lin contributed reagents and analysis tools; S.-C.H. Horng wrote the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the National Science and Technology Council in Taiwan, under Grant MOST111-2221-E-324-021.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Nomenclature

$\mathbf{x} = [x_1, \ldots, x_J]^T$	A design vector
$h(\mathbf{x})$	Deterministic cost function
$E[g_i(\mathbf{x})]$	The expectations of the ith constrained function
I	Number of constraints (unit)
d_i	Pre-specified requirement values
$\mathbf{Y} = [Y_1, \ldots, Y_J]^T$	Lower bound
$\mathbf{U} = [U_1, \ldots, U_J]^T$	Upper bound
$\overline{g}_i(\mathbf{x})$	Sample mean
L	Number of replications (unit)
$g_i^{\ell}(\mathbf{x})$	Estimation of the ℓth replication
η	Penalty factor
$f(\mathbf{x})$	Penalized cost function
$pe_i(\mathbf{x})$	Quadratic penalty function
L_a	The replications of the exact evaluation (unit)
$f_a(\mathbf{x})$	Penalized cost function through an exact evaluation
P	The number of PCE terms (unit)
w_p	Expansion coefficients
$\Phi_p(\hat{\mathbf{x}})$	Multivariate orthogonal polynomial basis functions
$H_p(\cdot)$	Hermite polynomials
Π	Number of training samples (unit)
$\mathbf{\Phi}$	Mapping vector of the expansion coefficients
B_f	Balance factor between exploration and exploitation
W_f	Probability of whale fall (percentage)
C_f	Jump strength of Levy flight
Ψ	Total number of beluga whales (unit)
$t_{\max}$	Maximum number of iterations (unit)
$\mathbf{x}_i^t = [x_{i,1}^t, \ldots, x_{i,J}^t]^{\mathrm{T}}$	The position of the ith beluga whale at iteration t
$\mathbf{r}_i^t = [r_{i,1}^t, \ldots, r_{i,J}^t]^{\mathrm{T}}$	The position of a randomly selected beluga whale at iteration t
$\mathbf{x}^* = [x_1^*, \ldots, x_J^*]^{\mathrm{T}}$	The position of the elite beluga whale
$B_{f\_\min}$	The lower bound of B_f
$B_{f\_\max}$	The upper bound of B_f
$W_{f\_\min}$	The lower bound of W_f
$W_{f\_\max}$	The upper bound of W_f
$C_{f\_\min}$	The lower bound of C_f
$C_{f\_\max}$	The upper bound of C_f
L_F	Levy flight function
C_a	The available computational effort (units)
N	Number of candidates (unit)
L_0	The essential replications (units)
L_n	The replications allocated to the nth candidate (units)
Δ	A one-time incremental computational effort

τ	A speed-up factor
$\hat{f}_n^{l+1}$	Incremental mean
$\hat{\delta}_n^{l+1}$	Incremental standard deviation
$\overline{f}_n^{l+1}$	Updated mean for overall replications
δ_n^{l+1}	The updated standard deviation for overall replications
$\lambda(t)$	The arrival interval rate of a patient (1/unit time)
$\mathbf{x} = [x_1, \ldots, x_5]^T$	A design vector
$E[g_1(\mathbf{x})]$	Average waiting time of critical patients (time unit)
$E[g_2(\mathbf{x})]$	Average waiting time of treatment patients (time unit)
Π	Number of randomly chosen samples (unit)
Ω	A representative subset
r	The rank of a superior design in Ω

References

1. Ta, T.A.; Mai, T.; Bastin, F.; L'Ecuyer, P. On a multistage discrete stochastic optimization problem with stochastic constraints and nested sampling. *Math. Program.* **2021**, *190*, 1–37. [CrossRef]
2. Lu, X.N.; Peng, Z.L.; Zhang, Q.; Yang, S.L. Event-based optimization approach for solving stochastic decision problems with probabilistic constraint. *Optim. Lett.* **2021**, *15*, 569–590. [CrossRef]
3. Latour, A.L.D.; Babaki, B.; Fokkinga, D.; Anastacio, M.H.; Hoos, H.H.; Nijssen, S. Exact stochastic constraint optimisation with applications in network analysis. *Artif. Intell.* **2022**, *304*, 103650. [CrossRef]
4. Ho, Y.C.; Zhao, Q.C.; Jia, Q.S. *Ordinal Optimization: Soft Optimization for Hard Problems*; Springer: New York, NY, USA, 2007.
5. Long, T.; Jia, Q.S.; Wang, G.M.; Yang, Y. Efficient real-time EV charging scheduling via ordinal optimization. *IEEE Trans. Smart Grid* **2021**, *2*, 4029–4038. [CrossRef]
6. Horng, S.C.; Lee, C.T. Integration of ordinal optimization with ant lion optimization for solving the computationally expensive simulation optimization problems. *Appl. Sci.* **2021**, *11*, 136. [CrossRef]
7. Horng, S.C.; Lin, S.S. Coupling elephant herding with ordinal optimization for solving the stochastic inequality constrained optimization problems. *Appl. Sci.* **2020**, *10*, 2075. [CrossRef]
8. Horng, S.C.; Lin, S.S. Ordinal optimization to optimize the job-shop scheduling under uncertain processing times. *Arab. J. Sci. Eng.* **2022**, *47*, 9659–9671. [CrossRef]
9. Horng, S.C.; Lin, S.S. Incorporate seagull optimization into ordinal optimization for solving the constrained binary simulation optimization problems. *J. Supercomput.* **2023**, *79*, 5730–5758. [CrossRef]
10. Liu, Y.; Zhao, G.; Li, G.; He, W.X.; Zhong, C.T. Analytical robust design optimization based on a hybrid surrogate model by combining polynomial chaos expansion and Gaussian kernel. *Struct. Multidiscip. Optim.* **2022**, *65*, 335. [CrossRef]
11. Yao, W.; Zheng, X.H.; Zhang, J.; Wang, N.G.; Tang, G.J. Deep adaptive arbitrary polynomial chaos expansion: A mini-data-driven semi-supervised method for uncertainty quantification. *Reliab. Eng. Syst. Saf.* **2023**, *229*, 108813. [CrossRef]
12. Geiersbach, C.; Loayza-Romero, E.; Welker, K. Stochastic approximation for optimization in shape spaces. *SIAM J. Optim.* **2021**, *31*, 348–376. [CrossRef]
13. Zhou, X.J.; Wang, X.Y.; Huang, T.W.; Yang, C.H. Hybrid intelligence assisted sample average approximation method for chance constrained dynamic optimization. *IEEE Trans. Industr. Inform.* **2021**, *17*, 6409–6418. [CrossRef]
14. Yu, C.L.; Lahrichi, N.; Matta, A. Optimal budget allocation policy for tabu search in stochastic simulation optimization. *Comput. Oper. Res.* **2023**, *150*, 106046. [CrossRef]
15. Cheng, D.L. Water allocation optimization and environmental planning with simulated annealing algorithms. *Math. Probl. Eng.* **2022**, *2022*, 2281856. [CrossRef]
16. Zhang, Q.B.; Yang, S.X.; Liu, M.; Liu, J.X.; Jiang, L. A new crossover mechanism for genetic algorithms for Steiner tree optimization. *IEEE Trans. Cybern.* **2022**, *52*, 3147–3158. [CrossRef]
17. Xu, H.Q.; Gu, S.; Fan, Y.C.; Li, X.S.; Zhao, Y.F.; Zhao, J.; Wang, J.J. A strategy learning framework for particle swarm optimization algorithm. *Inf. Sci.* **2023**, *619*, 126–152. [CrossRef]
18. Wang, Y.; Liu, Z.; Wang, G.G. Improved differential evolution using two-stage mutation strategy for multimodal multi-objective optimization. *Swarm Evol. Comput.* **2023**, *78*, 101232. [CrossRef]
19. Daneshyar, S.A.; Charkari, N.M. Biogeography based optimization method for robust visual object tracking. *Appl. Soft Comput.* **2022**, *122*, 108802. [CrossRef]
20. Beccaria, M.; Niccolai, A.; Zich, R.E.; Pirinoli, P. Shaped-beam reflectarray design by means of social network optimization (SNO). *Electronics* **2021**, *10*, 744. [CrossRef]
21. Tang, J.; Liu, G.; Pan, Q.T. A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 1627–1643. [CrossRef]
22. Chopraa, N.; Ansarib, M.M. Golden jackal optimization: A novel nature-inspired optimizer for engineering applications. *Expert Syst. Appl.* **2022**, *198*, 116924. [CrossRef]

23. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. Starling murmuration optimizer: A novel bio-inspired algorithm for global and engineering optimization. *Comput. Methods Appl. Mech. Eng.* **2022**, *392*, 114616. [CrossRef]
24. Braik, M.; Hammouri, A.; Atwan, J.; Al-Betar, M.A.A.; Awadallah, M.A. White shark optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. *Knowl.-Based Syst.* **2022**, *243*, 18457. [CrossRef]
25. Zhao, S.J.; Zhang, T.R.; Ma, S.L.; Chen, M. Dandelion Optimizer: A nature-inspired metaheuristic algorithm for engineering applications. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105075. [CrossRef]
26. Ahwazian, A.; Amindoust, A.; Tavakkoli-Moghaddam, R.; Nikbakht, M. Search in forest optimizer: A bioinspired metaheuristic algorithm for global optimization problems. *Soft Comput.* **2022**, *26*, 2325–2356. [CrossRef]
27. Hashim, F.A.; Hussien, A.G. Snake Optimizer: A novel meta-heuristic optimization algorithm. *Knowl.-Based Syst.* **2022**, *242*, 108320. [CrossRef]
28. Zhong, C.T.; Li, G.; Meng, Z. Beluga whale optimization: A novel nature-inspired metaheuristic algorithm. *Knowl.-Based Syst.* **2022**, *251*, 109215. [CrossRef]
29. Sasanfar, S.; Bagherpour, M.; Moatari-Kazerouni, A. Improving emergency departments: Simulation-based optimization of patients waiting time and staff allocation in an Iranian hospital. *Int. J. Healthc. Manag.* **2021**, *14*, 1449–1456. [CrossRef]
30. Wang, H.X.; Xie, F.; Li, J.; Miu, F. Modelling, simulation and optimisation of medical enterprise warehousing process based on FlexSim model and greedy algorithm. *Int. J. Bio-Inspired Comput.* **2022**, *19*, 59–66. [CrossRef]
31. Meng, Y.Z.; Chen, R.R.; Deng, T.H. Two-stage robust optimization of power cost minimization problem in gunbarrel natural gas networks by approximate dynamic programming. *Pet. Sci.* **2022**, *19*, 2497–2517. [CrossRef]
32. Dey, B.K.; Seok, H. Intelligent inventory management with autonomation and service strategy. *J. Intell. Manuf.* **2022**. [CrossRef] [PubMed]
33. Estrin, R.; Friedlander, M.P.; Orban, D.; Saunders, M.A. Implementing a smooth exact penalty function for equality-constrained nonlinear optimization. *SIAM J. Sci. Comput.* **2020**, *42*, A1809–A1835. [CrossRef]
34. Uemoto, T.; Naito, K. Support vector regression with penalized likelihood. *Comput. Stat. Data Anal.* **2022**, *174*, 107522. [CrossRef]
35. Zuo, Q.L. Settlement prediction of the piles socketed into rock using multivariate adaptive regression splines. *J. Appl. Sci. Eng.* **2023**, *26*, 111–119.
36. Zou, W.D.; Xia, Y.Q.; Cao, W.P. Back-propagation extreme learning machine. *Soft Comput.* **2022**, *26*, 9179–9188. [CrossRef]
37. Huang, S.H.; Mahmud, K.; Chen, C.J. Meaningful trend in climate time series: A discussion based on linear and smoothing techniques for drought analysis in Taiwan. *Atmosphere* **2022**, *13*, 444. [CrossRef]
38. Chen, C.H.; Lee, L.H. *Stochastic Simulation Optimization: An Optimal Computing Budget Allocation*; World Scientific: New Jersey, NJ, USA, 2010.
39. Yaseri, A.; Maghami, M.H.; Radmehr, M. A four-stage yield optimization technique for analog integrated circuits using optimal computational effort allocation and evolutionary algorithms. *IET Comput. Digit. Tech.* **2022**, *16*, 183–195. [CrossRef]
40. Chiu, C.C.; Lin, J.T. An efficient elite-based simulation-optimization approach for stochastic resource allocation problems in manufacturing and service systems. *Asia-Pac. J. Oper. Res.* **2022**, *39*, 2150030. [CrossRef]
41. Ryan, T.P. *Sample Size Determination and Power*; John Wiley and Sons: New Jersey, NJ, USA, 2013.
42. Al-Ebbini, L.M.K. An efficient allocation for lung transplantation using ant colony optimization. *Intell. Autom. Soft Comput.* **2021**, *35*, 1971–1985. [CrossRef]
43. Wang, Y.; Li, T.; Liu, X.J.; Yao, J. An adaptive clonal selection algorithm with multiple differential evolution strategies. *Inf. Sci.* **2022**, *604*, 142–169. [CrossRef]
44. Chakraborty, S.; Sharma, S.; Saha, A.K.; Saha, A. A novel improved whale optimization algorithm to solve numerical optimization and real-world applications. *Artif. Intell. Rev.* **2022**, *55*, 4605–4716. [CrossRef]
45. Amroune, M. Wind integrated optimal power flow considering power losses, voltage deviation, and emission using equilibrium optimization algorithm. *Energy Ecol. Environ.* **2022**, *7*, 369–392. [CrossRef]