processes

MDPI

Article

Reactive Power Optimization of Large-Scale Power Systems: A Transfer Bees Optimizer Application

Huazhen Cao [1], Tao Yu [2], Xiaoshun Zhang [3], Bo Yang [4,*] and Yaxiong Wu [1]

1 Power Grid Planning Research Center of Guangdong Power Grid Co., Ltd., Guangzhou 510062, China; zzw572@126.com (H.C.); jxpxlxwyx@163.com (Y.W.)
2 College of Electric Power, South China University of Technology, Guangzhou 510640, China; taoyu1@scut.edu.cn
3 College of Engineering, Shantou University, Shantou 515063, China; xszhang1990@sina.cn
4 Faculty of Electric Power Engineering, Kunming University of Science and Technology, Kunming 650500, China
* Correspondence: yangbo_ac@outlook.com; Tel.: +86-183-1459-6103

Received: 6 May 2019; Accepted: 24 May 2019; Published: 31 May 2019

Abstract: A novel transfer bees optimizer for reactive power optimization in a high-power system was developed in this paper. Q-learning was adopted to construct the learning mode of bees, improving the intelligence of bees through task division and cooperation. Behavior transfer was introduced, and prior knowledge of the source task was used to process the new task according to its similarity to the source task, so as to accelerate the convergence of the transfer bees optimizer. Moreover, the solution space was decomposed into multiple low-dimensional solution spaces via associated state-action chains. The transfer bees optimizer performance of reactive power optimization was assessed, while simulation results showed that the convergence of the proposed algorithm was more stable and faster, and the algorithm was about 4 to 68 times faster than the traditional artificial intelligence algorithms.

Keywords: transfer bees optimizer; reinforcement learning; behavior transfer; state-action chains; reactive power optimization

1. Introduction

Nonlinear programming is a very common issue in the operation of power systems, including reactive power optimization (RPO) [1], unit commitment (UC) [2], economic dispatch (ED) [3]. In order to tackle this issue, several optimization approaches have been adopted, such as the Newton method [4], quadratic programming [5], interior-point method [6]. However, these methods are essentially gradient-based mathematic optimization methods, which highly depend on an accurate system model. When there is nonlinearity, there are discontinuous functions and constraints, and there usually exist many local minimum upon which the algorithm may be easily fall into one local optimum [7].

In the past decades, artificial intelligence (AI) [8–18] has been widely used as an effective alternative because of its high independence from an accurate system model and strong global optimization ability. Inspired by nectar gathering of bees in wild nature, the artificial bee colony (ABC) [19] has been applied to optimal distributed generation allocation [8], global maximum power point (GMPP) tracking [20], multi-objective UC [21], and so on, and has the merits of simple structure, high robustness, strong universality, and efficient local search.

However, the ABC mainly depends on a simple collective intelligence without self-learning or knowledge transfer, which is a common weakness of AI algorithms such as genetic algorithm (GA) [9], particle swarm optimization (PSO) [10], group search optimizer (GSO) [11], ant colony

system (ACS) [12], interactive teaching–learning optimizer (ITLO) [13], grouped grey wolf optimizer (GGWO) [14], memetic salp swarm algorithm (MSSA) [15], dynamic leader-based collective intelligence (DLCI) [16], and evolutionary algorithms (EA) [17]. Thus, a relatively low search efficiency may result, particularly while considering a new optimization task of a complex industrial system [22], e.g., the optimization of a large-scale power system with different complex new tasks. In fact, the computational time of these algorithms can be effectively reduced for RPO or optimal power flow (OPF) via the external equivalents in some areas (e.g., distribution networks) [23]. This is because the optimization scale and difficulty are significantly reduced as the number of optimization variables and constraints decreases. However, the optimization results are highly determined by the accuracy of the external equivalent model [24], which is usually worse than that obtained by global optimization. Hence, this paper aims to propose a fast and efficient AI algorithm for global optimization.

Previous studies [25] discovered that bees have evolved an instinct to memorize the beneficial weather conditions of their favorite flowers, e.g., temperature, air humidity, and illumination intensity, which may rapidly guide bees to find the best nectar source in a new environment with high probability, hence, the survival and prosperity of the whole species living in various environments can be guaranteed via the exploitation of such knowledge. The above natural phenomenon resulted from the struggle for existence in a harsh and unknown environment can be regarded as a knowledge transfer, which has been popularly investigated in machine learning and data mining [26]. In practice, prior knowledge is from the source tasks and then it is applied to a new but analogous assignment, such that fewer training data can be used to achieve a higher learning efficiency [27], e.g., learn to ride a bike before starting to ride a motorcycle. In fact, knowledge transfer-based optimization is essentially the knowledge-based history data-driven method [28], which can accelerate the optimization speed of a new task according to prior knowledge. Also, reinforcement learning (RL) can be accelerated by knowledge conversion [29], and agents learn new tasks faster and interact less with the environment. As a consequence, knowledge transfer reinforcement learning (KTRL) has been developed [30] through combining AI and behavior psychology [31] and is divided into behavior shift and information shift.

In this paper, behavior shift was used for Q-learning [32] to accelerate the learning of a new task, which was called $\boldsymbol{Q}$-value transfer. The $\boldsymbol{Q}$-value matrix was applied in knowledge learning, storage, and transfer. However, the practical application of conventional Q-learning was restricted to only a group of new tasks with small size due to the calculation burden. To deal with this obstacle, an associated state-action chain was introduced after the solution space was decomposed into several low-dimensional solution spaces. Therefore, this paper proposes a transfer bee optimizer (TBO) based on Q-learning and behavior transfer. The main novelties and contributions of this work are given as follows:

(1) Compared with the traditional mathematic optimization methods, the TBO has a stronger global search ability by employing the scouts and workers for exploitation and exploration. Besides, it can approximate the global optimum more closely via global optimization instead of external equivalent-based local optimization.
(2) Compared with the general AI algorithms, the TBO can effectively avoid a blind search in the initial optimization phase and implement a much faster optimization for a new task by utilizing prior knowledge from the source tasks.
(3) The optimization performance of the TBO was thoroughly tested by RPO of large-scale power systems. Because of its high optimization flexibility and superior optimization efficiency, it can be extended to other complex optimization problems.

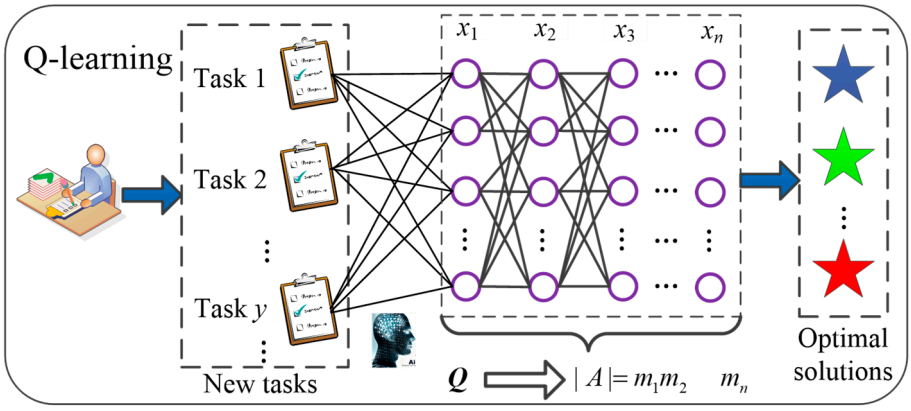
The remaining of this paper is arranged as follows: Section 2 presents the basic principles of the TBO. Section 3 designs the TBO for RPO. Section 4 shows the simulation results, and Section 5 summarizes paper.

2. Transfer Bees Optimizer

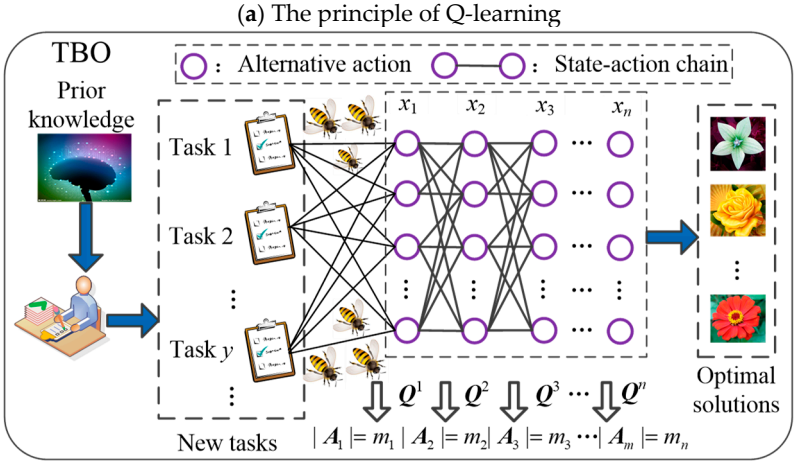
2.1. State-Action Chain

Q-learning typically finds and learns different state-action pairs through a look-up table, i.e., $\boldsymbol{Q}(s,a)$, but this is not enough to handle a complex task with multiple controllable variables because of the curse of dimension, as illustrated in Figure 1a. Suppose the optional operand of the controllable variable x_i is m_i, and $|A| = m_1 m_2 \cdots m_n$, n is the sum of the controlled variables, and A is the action set. If n is dramatically increased, the dimension of $\boldsymbol{Q}$-value matrix will grow very fast, so the calculation convergence is slow and may even lead to failure. Hierarchical reinforcement learning (HRL) [33] is commonly used to avert this obstacle and decomposes the original complex task into several simpler subtasks, e.g., MAXQ [34]. However, it is easy to fall into a near-optimum for the overall task due to the fact that it is difficult to design and coordinate all the subtasks.

In contrast, the whole solution space is decomposed into several low-dimensional solution spaces by the associated state-action chain. In such frame, each controlled variable has a unique memory matrix $\boldsymbol{Q}^i$, the size of $\boldsymbol{Q}$-value matrix can be significantly reduced, it has the advantages of convenient storage and transfer, and the controlled variables are linked, as shown in Figure 1b.



(**a**) The principle of Q-learning



(**b**) The principle of TBO

Figure 1. Comparison of Q-learning and transfer bee optimizer (TBO).

2.2. Knowledge Learning and Behavior Transfer

2.2.1. Knowledge learning

Figure 2 shows that all bees search a nectar source according to their prior knowledge ($\boldsymbol{Q}$-value matrix); the obtained knowledge will then be updated after each interaction with the nectar source, therefore a cycle of knowledge learning and conscious exploration can be fully completed. As shown in

Figure 1a, a simple RL agent is usually adopted for traditional Q-learning to acquire knowledge [18,35], which is the cause of inefficient learning. In contrast, the TBO adopts the method of swarm collaborative exploration for knowledge learning, which can update multiple elements of the Q value matrix at the same time, thus greatly improving the learning efficiency, which can be described as [32]

$$\begin{aligned} Q_{k+1}^{i}(s_k^{ij}, a_k^{ij}) = Q_k^{i}(s_k^{ij}, a_k^{ij}) + \alpha[R^{ij}(s_k^{ij}, s_{k+1}^{ij}, a_k^{ij}) + \\ \gamma \max_{a^i \in A_i} Q_k^{i}(s_{k+1}^{ij}, a) - Q_k^{i}(s_k^{ij}, a_k^{ij})], \\ j = 1, 2, \cdots, J; i = 1, 2, \cdots, n \end{aligned} \tag{1}$$

where α represents the factor of knowledge study; γ is the discount coefficient; the superscripts i and j signify the ith Q-value matrix (i.e., the ith controlled variable) and the jth bee, respectively; J is the population size of bees; (s_k,a_k) means a pair of state-action in the kth iteration; $R(s_k,s_{k+1},a_k)$ is the reward function that transforms from state s_k to s_{k+1} used under an optional operation a_k; a^i means random optional action of the ith controlled variable x_i; and A_i represents the multiple active result sets of the ith controlled variable.

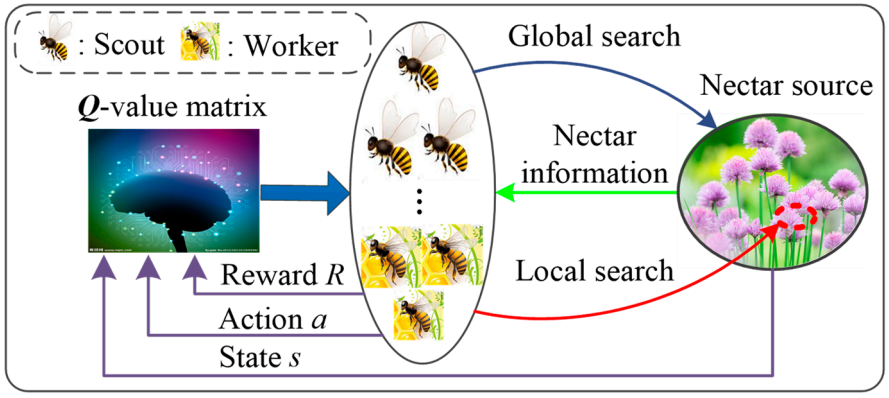


Figure 2. The principle of knowledge learning of the TBO inspired by the nectar gathering of bees.

2.2.2. Behavior transfer

In the initial process, the TBO needs to go through a series of source tasks to get the optimal Q-value matrix, so as to make use of and to prepare prior knowledge for similar new tasks in the future. The relevant prior knowledge of source task is shown in Figure 3. According to the similarity of the source task, the optimal Q-value matrix of the source task Q_S^* is shifted from the initial Q-value matrix to the new task Q_N^0, as

$$Q_N^0 = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1h} & \cdots \\ r_{21} & r_{22} & \cdots & r_{2h} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \cdots \\ r_{y1} & r_{y2} & \cdots & r_{yh} & \cdots \\ \vdots & \vdots & \cdots & \vdots & \ddots \end{bmatrix} Q_S^* \tag{2}$$

with

$$Q_N^0 = \begin{bmatrix} Q_{N1}^{10} & \cdots & Q_{N1}^{i0} & \cdots & Q_{N1}^{n0} \\ Q_{N2}^{10} & \cdots & Q_{N2}^{i0} & \cdots & Q_{N2}^{n0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ Q_{Ny}^{10} & \cdots & Q_{Ny}^{i0} & \cdots & Q_{Ny}^{n0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}, Q_S^* = \begin{bmatrix} Q_{S1}^{1*} & \cdots & Q_{S1}^{i*} & \cdots & Q_{S1}^{n*} \\ Q_{S2}^{1*} & \cdots & Q_{S2}^{i*} & \cdots & Q_{S2}^{n*} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ Q_{Sh}^{1*} & \cdots & Q_{Sh}^{i*} & \cdots & Q_{Sh}^{n*} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

where Q^{i0}_{Ny} is the ith initial Q-value matrix in the yth new task; Q^{i*}_{Sh} is the ith optimized Q-value matrix in the hth source task; and r_{yh} represents the comparability between the hth source task and the yth new task; here, a large r_{yh} indicates that the yth new task can gain much knowledge from the hth source task, and $0 \le r_{yh} \le 1$.

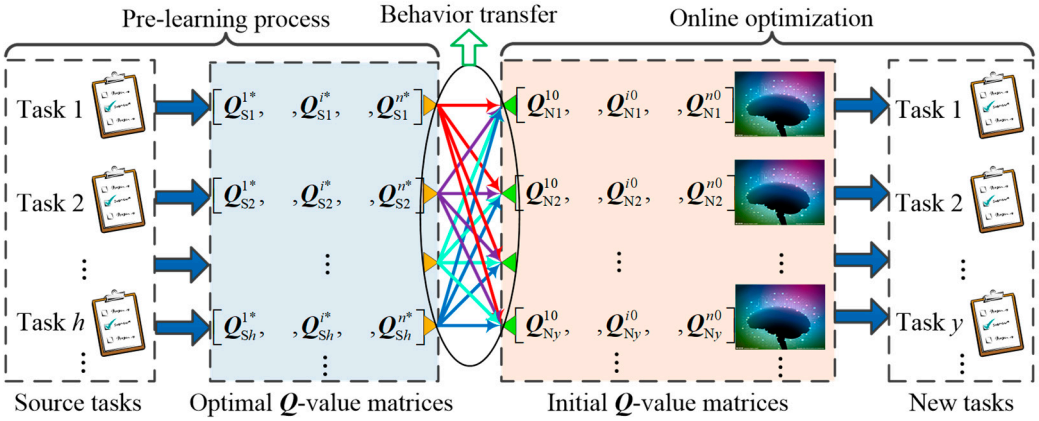


Figure 3. Behavior transfer of the TBO.

2.3. Exploration and Feedback

2.3.1. Action policy

There are two kinds of bees in Figure 2, e.g., scout and worker, determined by their nectar amounts (fitness values) [19], which are responsible for global searching and local searching. As a consequence, a bee colony can balance the exploration and exploitation through different action policies in a nectar source. In the TBO, 50% of bees with nectar amounts that rank in the half top of all bees are designated as worker, while the others are scout. On the basis of the ε-*Greedy* rule [31], the scouts' actions are based on the proportion of Q-value in the current status. As for the controlled variable x_i, one behavior of each scout is chosen as

$$a^{ij}_{k+1} = \begin{cases} \underset{a^i \in A_i}{\operatorname{argmax}} Q^i_{k+1}(s^{ij}_{k+1}, a^i), & \text{if } \varepsilon \le \varepsilon_0 \\ a_{\mathrm{rg}}, & \text{otherwise} \end{cases} \tag{3}$$

where ε is any value, uniformly distributed between [0, 1]; ε_0 represents the exploration rate; and a_{rg} represents any global behavior ascertained by the distribution of the state-action probability matrix $\boldsymbol{P}^i$, updated by

$$\begin{cases} \boldsymbol{P}^i(s^i, a^i) = \dfrac{\boldsymbol{e}^i(s^i, a^i)}{\sum\limits_{a' \in A_i} \boldsymbol{e}^i(s^i, a')} \\ \boldsymbol{e}^i(s^i, a^i) = \dfrac{1}{\max\limits_{a' \in A_i} \boldsymbol{Q}^i(s^i, a') - \beta \boldsymbol{Q}^i(s^i, a^i)} \end{cases} \tag{4}$$

where β is the discrepancy factor, and $\boldsymbol{e}^i$ is an evaluation matrix of the pairs of the state-action.

On the other hand, the workers keep exploring new nectar sources at nearby nectar sources, which can be written as [8]

$$\begin{cases} a^{ij}_{\mathrm{new}} = a^{ij} + \mathrm{Rnd}(0,1)(a^{ij} - a^{ij}_{\mathrm{rand}}) \\ a^{ij}_{\mathrm{rand}} = a^i_{\min} + \mathrm{Rnd}(0,1)(a^i_{\max} - a^i_{\min}) \end{cases} \tag{5}$$

where a^{ij}_{new}, a^{ij}, and a^{ij}_{rand} denote the new action, current action, and random action of the ith controlled variable selected by the jth worker; $a^i_{\max}$ and $a^i_{\min}$ are the maximum and minimum behavior, respectively, in the ith controlled variable.

2.3.2. Reward function

After each exploration, each bee will get an instant reward based on its fitness value. Because it is the goal of the TBO to maximize the expected long-term rewards for each state [28], the reward function is designed as

$$R^{ij}(s_k^{ij}, s_{k+1}^{ij}, a_k^{ij}) = \frac{C}{f_k^j} \tag{6}$$

where C is a positive multiplicator, and $f_k{}^j$ represents the fitness function of the jth bee in the kth iteration. This is closely related to the target function.

After each bee obtains its new reward, the scouts and workers will swap their roles according to the obtained reward rank, precisely, 50% of bees who had a larger reward will become workers. As a result, a compromise is reached between exploration and development to ensure a global search for the TBO.

3. Design of the TBO for RPO

3.1. Mathematical Model of RPO

As the subproblem of OPF, the conventional RPO aims to lower the active power losses or other appropriate target functions via optimizing the different types of controlled variables (e.g., transformer tap ratio) under multiple equality constraints and inequality constraints [35]. In this article, the integrated target function of the active power loss and the deviation of supply voltage were used as follows [18]

$$\min f(\boldsymbol{x}) = \mu P_{\text{loss}} + (1-\mu) V_{\text{d}} \tag{7}$$

subject to

$$\begin{cases} P_{\text{G}i} - P_{\text{D}i} - V_i \sum\limits_{j \in N_i} V_j (g_{ij} \cos\theta_{ij} + b_{ij} \sin\theta_{ij}) = 0, i \in N_0 \\ Q_{\text{G}i} - Q_{\text{D}i} - V_i \sum\limits_{j \in N_i} V_j (g_{ij} \sin\theta_{ij} - b_{ij} \cos\theta_{ij}) = 0, i \in N_{\text{PQ}} \\ Q_{\text{G}i}^{\min} \le Q_{\text{G}i} \le Q_{\text{G}i}^{\max}, \quad i \in N_{\text{G}} \\ V_i^{\min} \le V_i \le V_i^{\max}, \quad i \in N_i \\ Q_{\text{C}i}^{\min} \le Q_{\text{C}i} \le Q_{\text{C}i}^{\max}, \quad i \in N_{\text{C}} \\ T_k^{\min} \le T_k \le T_k^{\max}, \quad k \in N_{\text{T}} \\ |S_l| \le S_l^{\max}, \quad l \in N_{\text{L}} \end{cases} \tag{8}$$

where $\boldsymbol{x} = [V_{\text{G}}, T_k, Q_{\text{C}}, V_{\text{L}}, Q_{\text{G}}]$ represents the variable vector, V_{G} represents the terminal voltage of generator; T_k means the transformer tapping ping ratio; Q_{C} is the reactive power of the shunt capacitor; V_{L} is the load-bus voltage; Q_{G} is the reactive power of the generator; P_{loss} is the power loss; V_{d} is the deviation of supply voltage; $0 \le \mu \le 1$ is the weight coefficient; $P_{\text{G}i}$ and $Q_{\text{G}i}$ are the generated active power; $P_{\text{D}i}$ and $Q_{\text{D}i}$ are the demanded active and reactive power, respectively; $Q_{\text{C}i}$ represents the reactive power compensation; V_i and V_j are the voltage magnitude of the ith and jth node, respectively; θ_{ij} is the phase difference of voltage; g_{ij} represents the conductance in the transmission line i-j; b_{ij} represents the susceptance of the transmission line i-j; S_l is the apparent power of the transmission line l; N_i is the node set; N_0 is the set of the slack bus; N_{PQ} is the set of active/reactive power (PQ) buses; N_{G} is the unit set; N_{C} is the compensation equipment; N_{T} is the set of transformer taps; and N_{L} is the branch set. In addition, the active power loss and the deviation of supply voltage can be computed by [18]

$$P_{\text{loss}} = \sum_{i,j \in N_L} g_{ij} [V_i^2 + V_j^2 - 2V_i V_j \cos\theta_{ij}] \tag{9}$$

$$V_{\mathrm{d}} = \sum_{i \in N_i} \left| \frac{2V_i - V_i^{\max} - V_i^{\min}}{V_i^{\max} - V_i^{\min}} \right| \tag{10}$$

3.2. Design of the TBO

3.2.1. Design of state and action

The terminal voltage of generator, transformer tapping ratio, and d shunt capacitor reactive power compensation were chosen as the controlled variables of RPO, in which each controlled variable had its own $\boldsymbol{Q}$-value matrix $\boldsymbol{Q}^i$ and action set $\boldsymbol{A}_i$, as shown in Figure 1. In addition, the operation sets for every controlled variable were the state sets for the next controlled variable, i.e., $\boldsymbol{S}_{i+1} = \boldsymbol{A}_i$, where the initial controlled variable depends on different tasks of RPO, thus each task can be considered as a specific state of x_1.

3.2.2. Design of the reward function

It can be found from (6) that the fitness function determines the reward function that represents the overall target function (7) and needs to satisfy all constraints (8). Hence, the fitness function is designed as

$$f^j = \mu P_{\mathrm{loss}}^j + (1-\mu) V_{\mathrm{d}}^j + \eta q^j \tag{11}$$

where q represents the sum of inequalities that violate the constraints, and η represents the regularization factor.

3.2.3. Behavior transfer for RPO

Based on eq (2), the transfer efficiency of TBO mainly depends on getting the comparability between the source tasks and the new tasks [30]. In fact, the distribution of power flow principally determines the RPO in the power system, thus it is principally influenced by the power demand, because the topological structure of the power system cannot be changed much daily. Therefore, the active power demand was divided into several load intervals as follows

$$\left\{ \left[P_{\mathrm{D}}^1, P_{\mathrm{D}}^2\right), \left[P_{\mathrm{D}}^2, P_{\mathrm{D}}^3\right), \cdots, \left[P_{\mathrm{D}}^h, P_{\mathrm{D}}^{h+1}\right), \cdots, \left[P_{\mathrm{D}}^{H-1}, P_{\mathrm{D}}^H\right] \right\} \tag{12}$$

where P_{D}^h represents the demand of active power in the hth source task for RPO, $P_{\mathrm{D}}^1 < P_{\mathrm{D}}^2 < \cdots < P_{\mathrm{D}}^h < \cdots < P_{\mathrm{D}}^H$.

Suppose the active power required in the yth new task is P_{D}^y, $P_{\mathrm{D}}^1 < P_{\mathrm{D}}^y < P_{\mathrm{D}}^H$, then the comparability between two tasks will be computed by

$$r_{yh} = \frac{\left[W + \Delta P_{\mathrm{D}}^{\max}\right] - \left|P_D^h - P_D^y\right|}{\sum_{h=1}^{H} \left\{ \left[W + \Delta P_{\mathrm{D}}^{\max}\right] - \left|P_D^h - P_D^y\right| \right\}} \tag{13}$$

$$\Delta P_{\mathrm{D}}^{\max} = \max_{h=1,2,\cdots,H} \left(\left|P_D^h - P_D^y\right| \right) \tag{14}$$

where W represents the transfer factor, and $\Delta P_{\mathrm{D}}^{\max}$ represents the ultimate error of active power demand, where $\sum_{h=1}^{H} r_{yh} = 1$.

Note that a smaller deviation $\left|P_D^h - P_D^y\right|$ brings in a larger similarity r_{yh}, therefore the new yth task can develop more knowledge.

Therefore, the overall process of TBO behavior transfer is generalized as follows:

Step 1. Determine the source tasks according to a typical load curve in a day by Equation (12);
Step 2. Complete the source tasks in the initial study process and store their optimal $\boldsymbol{Q}$-value matrices;

Step 3. Calculate the comparability between original tasks and new task according to the deviation of power demand according to Equations (13) and (14);

Step 4. Obtain the original $\boldsymbol{Q}$-value matrices in the new task by Equation (2).

3.2.4. Parameters setting

For the TBO, eight parameters, α, γ, J, ε_0, β, C, η, and W are important and need to be set following the general guidelines below [18,26,32,35].

Of these, α represents the knowledge learning factor, with $0 < \alpha < 1$, a determines the rate of knowledge acquisition of the bees. A larger α will accelerate knowledge learning, which may bring about a local optimization, while a smaller value can improve the algorithm stability [35].

The discount factor is defined to exponentially discount the rewards obtained by the $\boldsymbol{Q}$-value matrix in the future, and its value is $0 < \gamma < 1$. Since the future return on RPO is insignificant, it is required assume a value close to zero [18].

J is the number of bees, with $J \geq 1$; it determines the rate of convergence and the rate of solution. A large J makes the algorithm approximate a more accurate global optimization solution but results in a larger computational burden [26].

The exploration rate, $0 < \varepsilon_0 < 1$, balances the exploration and development of a nectar resource by the scouts. The scouts are encouraged to pick a voracious action instead of any action according to the state-action probability matrix by a larger ε_0.

The discrepancy factor, $0 < \beta < 1$, increases the differences among the elements of each row in $\boldsymbol{Q}$-value matrices.

The positive multiplicator, $C > 0$, distributes the weight and fitness functions of the reward function. The bees are encouraged to get more rewards by the fitness function with a large C, while the difference in rewards is smaller among all bees.

The penalty factor, $\eta > 0$, makes sure to satisfy the restrain of inequality. Solution infeasibility may arise because of a smaller η [18]. Here, W is the shift factor, with $W \geq 0$, that identifies the comparability among two tasks.

The parameters were selected through trial and error, as shown in Table 1.

Table 1. Parameters used in TBO.

Parameter	Range	IEEE 118-Bus System		IEEE 300-Bus System	
		Pre-Learning	Online Optimization	Pre-Learning	Online Optimization
α	$0 < \alpha < 1$	0.95	0.99	0.9	0.95
γ	$0 < \gamma < 1$	0.2	0.2	0.3	0.3
J	$J \geq 1$	15	5	30	10
ε_0	$0 < \varepsilon_0 < 1$	0.9	0.98	0.95	0.98
β	$0 < \beta < 1$	0.95	0.95	0.98	0.98
C	$C > 0$	1	1	1	1
η	$\eta > 0$	10	10	50	50
W	$W \geq 0$	—	50	—	100

3.2.5. Execution Procedure of the TBO for RPO

At last, the overall implementation process of TBO is shown in Figure 4, and $\left\| Q_{k+1}^{i} - Q_{k}^{i} \right\|_2 < \sigma$ is the memory value difference of matrix 2-norm, with the precision factor $\sigma = 0.001$.

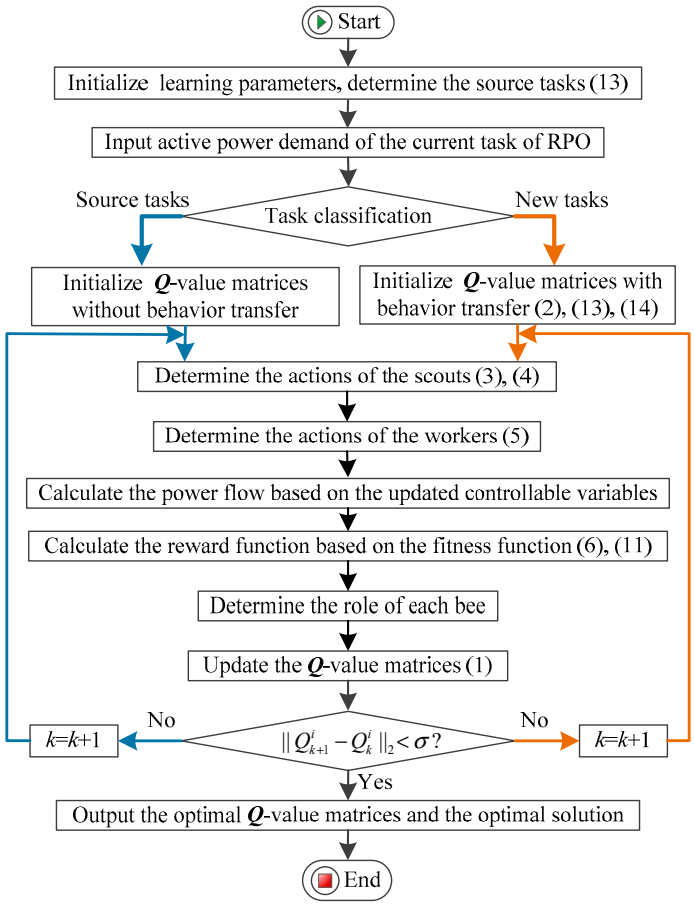


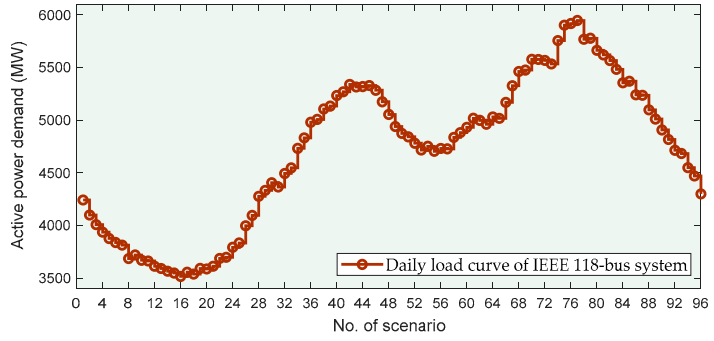
Figure 4. Flowchart of TBO for reactive power optimization (RPO).

4. Case Studies

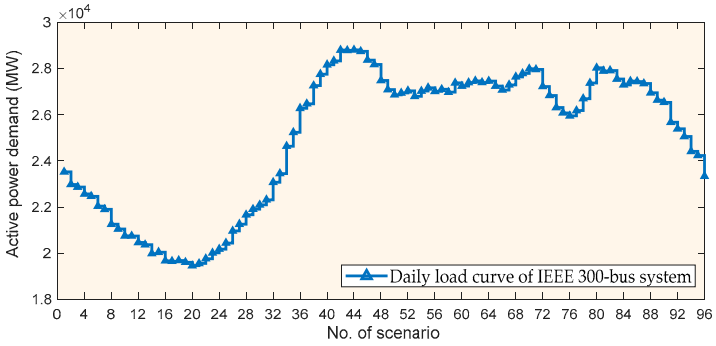
The TBO for RPO was assessed by the IEEE 118-bus system and the IEEE 300-bus system and compared with those of ABC [8], GSO [11], ACS [12], PSO [10], GA [9], quantum genetic algorithm (QGA) [36], and ant colony based Q-learning (Ant-Q) [37]. Furthermore, the main parameters of other algorithms were obtained through trial and error and were set according to reference [38], while the weight coefficient μ applied to eq (7) assigned the active power loss and the deviation of the output voltage. The simulation is executed on Matlab 7.10 by a personal computer with Intel(R) Xeon (R) E5-2670 v3 CPU at 2.3 GHz with 64 GB of RAM.

The IEEE 118-bus system consists of 54 generators and 186 branches and is divided into three voltage levels, i.e., 138 kV, 161 kV, and 345 kV. The IEEE 300-bus system is constituted by 69 generators and 411 branches, with 13 voltage levels, i.e., 0.6 kV, 2.3 kV, 6.6 kV, 13.8 kV, 16.5 kV, 20 kV, 27 kV, 66 kV, 86 kV, 115 kV, 138 kV, 230 kV, and 345 kV [39–41]. The number of controlled variables in IEEE 118-bus system was 25, and the number of controlled variables in IEEE 300-bus system was 111. More specifically, reactive power compensation is divided into five levels from rated level [−40%, −20%, 0%, 20%, 40%], the transformer tapping is divided into three levels [0.98, 1.00, 1.02], and the terminal voltage of generator is uniformly discretized into [1.00, 1.01, 1.02, 1.03, 1.04, 1.05, 1.06].

According to the given typical daily load curves shown in Figure 5, the active power demand was discretized into 20 and 22 load intervals, respectively, where every interval was 125 MW and 500 MW, respectively, i.e., {[3500, 3625), [3625, 3750), . . . , [5875, 6000]} MW and {[19,000, 19,500), [19,500, 20,000), . . . , [28,500, 29,000]} MW. Moreover, the implementation time of RPO was set at 15 min. Hence, the number of new tasks per day was 96, while the source tasks of the IEEE 118-bus system was 21, and the original tasks of the IEEE 300-bus system was 23.



(**a**) IEEE 118-bus system.

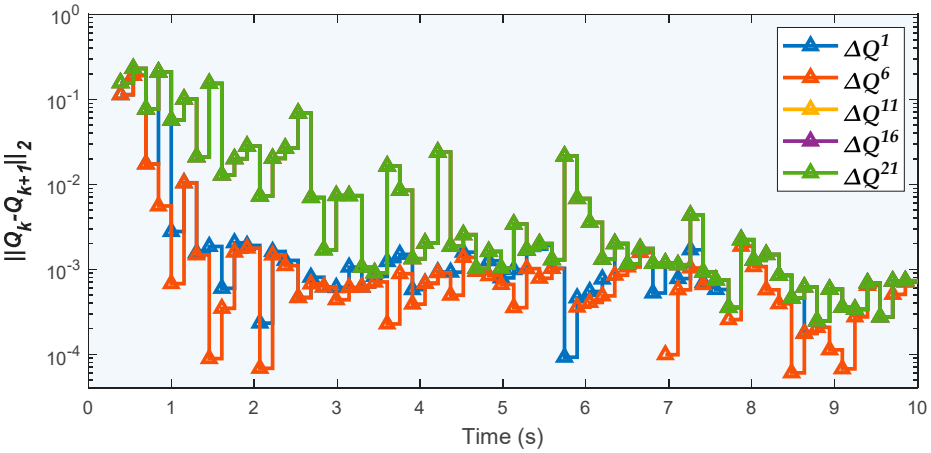


(**b**) IEEE 300-bus system

Figure 5. Typical daily load curves of the IEEE 118-bus system and IEEE 300-bus system.

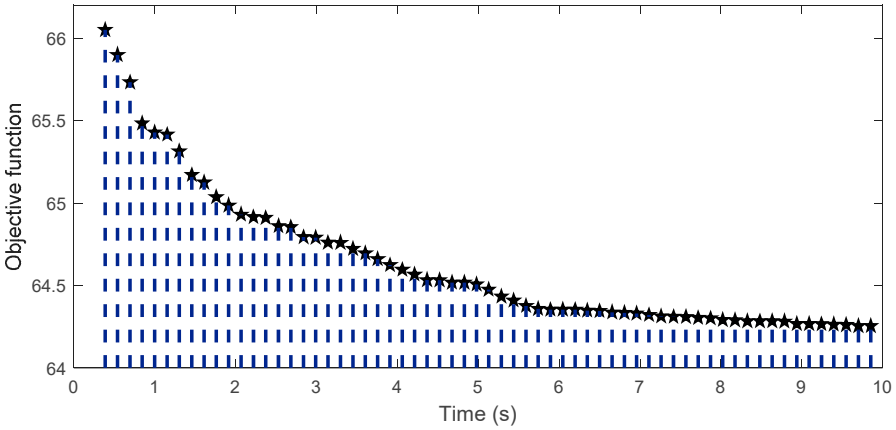
4.1. Study of the Pre-Learning Process

The TBO required a preliminary study to gain the optimal Q-value matrices for all source tasks, and then convert them to an initial Q-value. Figures 6 and 7 illustrate that the TBO will astringe to the optimal Q-value matrices of the source tasks while the optimal objective function can be obtained. Furthermore, the convergence of all Q-value matrices was consistent, as the same feedback rewards were used from the same bees to update the Q-value matrices at each iteration.



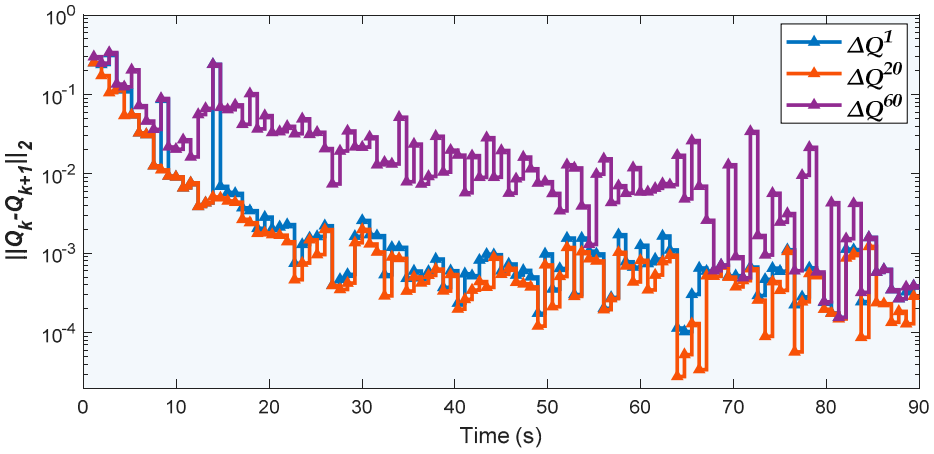
(**a**) Matrix 2-norms of Q-value differences

Figure 6. *Cont.*

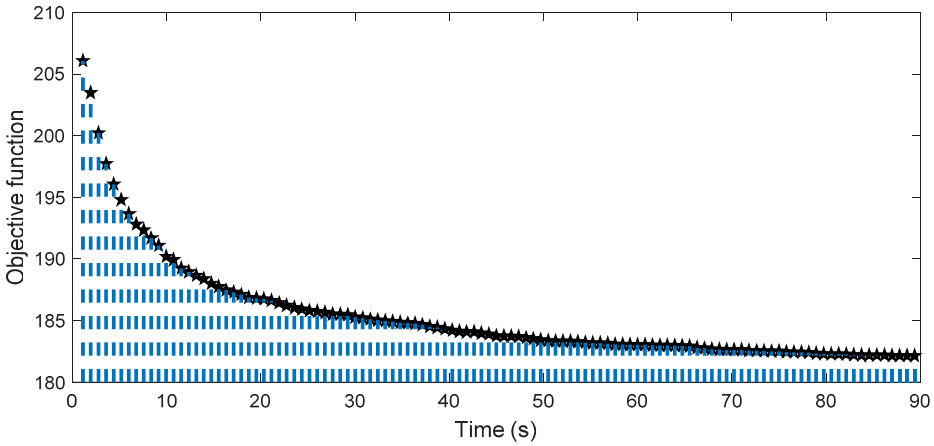


(**b**) Objective function

Figure 6. Convergence of the seventh source task of the IEEE 118-bus system obtained in the pre-learning process.



(**a**) Matrix 2-norms of Q-value differences



(**b**) Objective function

Figure 7. Convergence of the eighth source task of the IEEE 300-bus system obtained in the pre-learning process.

4.2. Study of Online Optimization

4.2.1. Study of behavior transfer

Through the preliminary study process, the TBO was ready for online optimization of RPO under different scenarios (different new tasks) with prior knowledge. The convergence of the target functions

gained by different algorithms in online optimization is given in Figures 8 and 9. Compared to the preliminary study process, the convergence of the TBO was approximately 10 to 20 times that of other algorithms in online optimization, which verified the effectiveness of knowledge transfer. Furthermore, the convergence rate of the TBO algorithm was much faster than that of other algorithms due to transcendental knowledge exploitation.

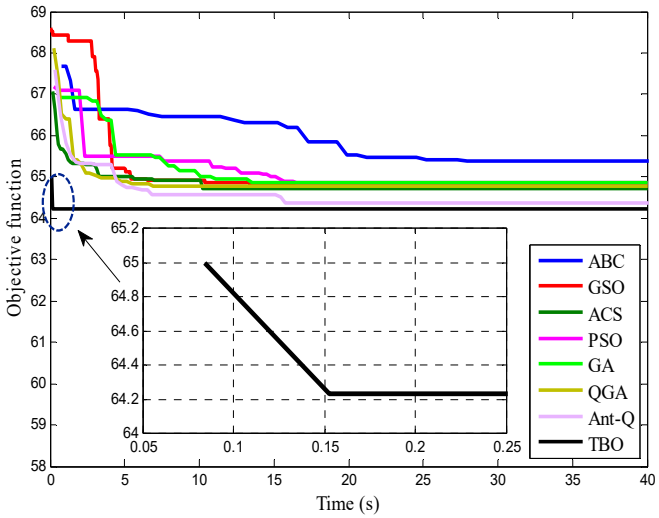


Figure 8. Convergence of the second new task of the IEEE 118-bus system obtained in the online optimization.

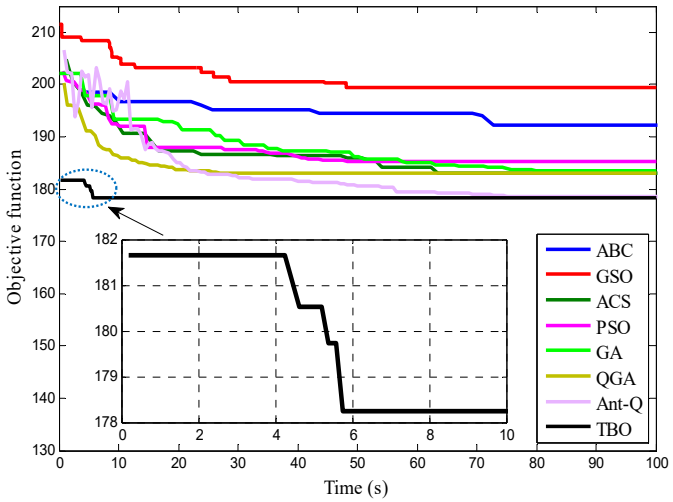
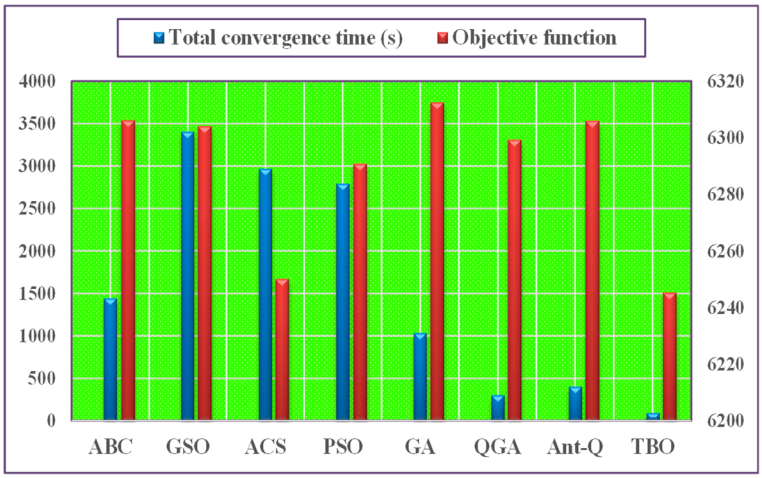


Figure 9. Convergence of the fourth new task of the IEEE 300-bus system obtained in the online optimization.

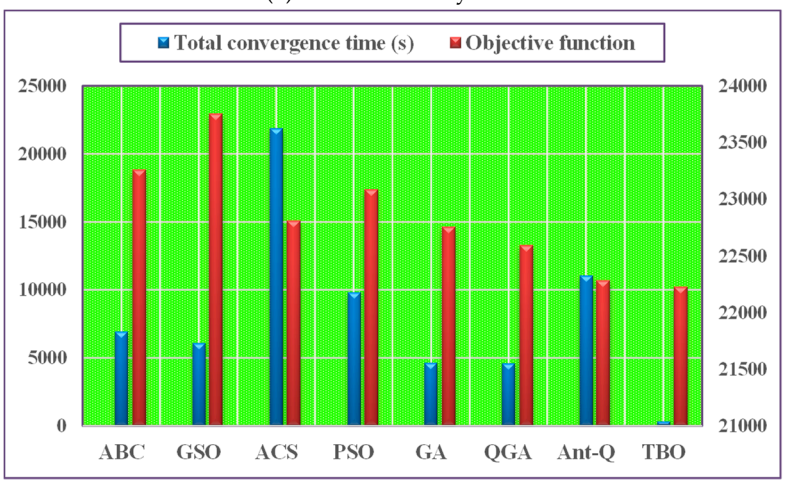
4.2.2. Comparative results and discussion

Tables 2 and 3 provide the performance and the statistical results gained by these algorithms in 10 runs, in which the convergence time was the average time of each scene, and the others were the average time of one day. The variance, standard deviation (Std. Dev.), and relative standard deviation (Rel. Std. Dev.) [42–44] were introduced in order to assess the stability. One can easily find that the convergence rate of the TBO was faster compared with that of other algorithms, as illustrated in Figure 10. Compared with that of other algorithms, the convergence rate of the TBO was about 4 to 68 times, indicating the benefit of cooperative exploration and action transfer. In addition, the optimization target function from the TBO was much smaller than that of other algorithms, which verified the advantageous effect of self-learning and global search. Note that the TBO would gain a

better solution which is closer to the global optimum with respect to other algorithms with a smaller optimal objective function in most new tasks (72.92% of new tasks on the IEEE 118-bus system and 89.58% of new tasks on the IEEE 300-bus system), as shown in Figures 11 and 12.



(**a**) IEEE 118-bus system



(**b**) IEEE 300-bus system

Figure 10. Comparison of performance of different algorithms obtained in 10 runs.

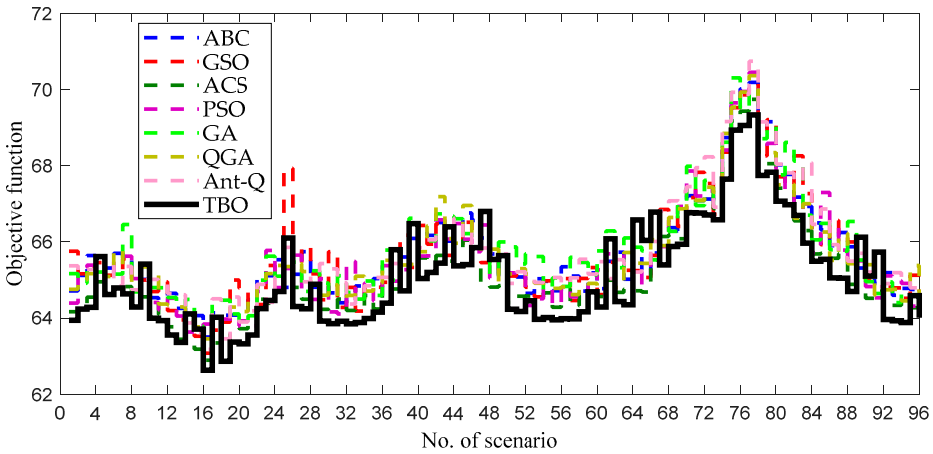


Figure 11. Optimal objective function of the IEEE 118-bus system obtained by different algorithms in 10 runs.

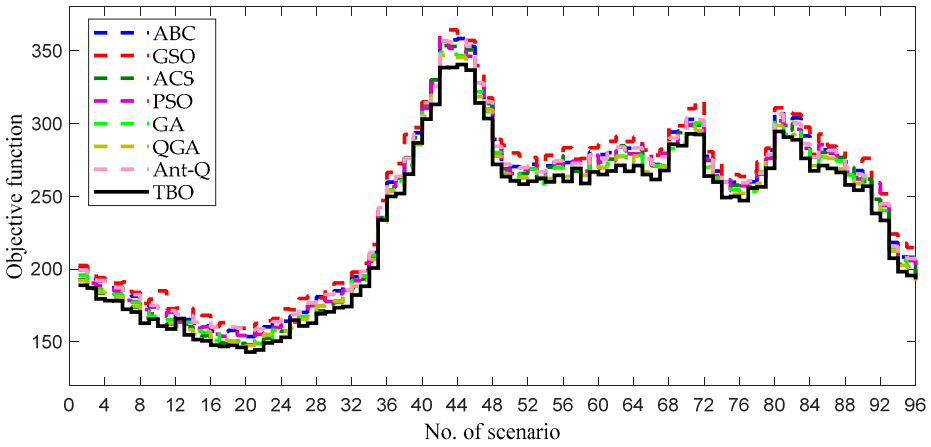


Figure 12. Optimal objective function of the IEEE 300-bus system obtained by different algorithms in 10 runs.

Table 2. Performance indices results of different algorithms on the IEEE 118-bus system obtained in 10 runs. ABC: artificial bee colony, GSO: group search optimizer, ACS: ant colony system, PSO: particle swarm optimization, GA: genetic algorithm, QGA: quantum genetic algorithm, Ant-Q: ant colony based Q-learning

Algorithm Index	ABC	GSO	ACS	PSO	GA	QGA	Ant-Q	TBO
Convergence time (s)	15	35.5	30.9	29.1	10.8	3.99	4.16	0.94
P_{loss} (MW)	1.11×10^4	1.11×10^4	1.11×10^4	1.11×10^4	1.11×10^4	1.11×10^4	1.11×10^4	1.10×10^4
V_{d} (%)	1.51×10^3	1.49×10^3	1.44×10^3	1.48×10^3	1.50×10^3	1.51×10^3	1.50×10^3	1.48×10^3
f	6.31×10^3	6.30×10^3	6.25×10^3	6.29×10^3	6.31×10^3	6.30×10^3	6.31×10^3	6.25×10^3
Best	6.30×10^3	6.30×10^3	6.24×10^3	6.28×10^3	6.31×10^3	6.30×10^3	6.30×10^3	6.24×10^3
Worst	6.31×10^3	6.31×10^3	6.25×10^3	6.30×10^3	6.32×10^3	6.30×10^3	6.31×10^3	6.25×10^3
Variance	4.02	19.3	6.43	16.3	12	6.37	8.57	2.27
Std. Dev.	2.01	4.39	2.54	4.03	3.46	2.52	2.93	1.51
Rel. Std. Dev	3.18×10^{-4}	6.96×10^{-4}	4.06×10^{-4}	6.41×10^{-4}	5.49×10^{-4}	4.01×10^{-4}	4.64×10^{-4}	2.41×10^{-4}

Table 3. Performance indices results of different algorithms on the IEEE 300-bus system obtained in 10 runs.

Algorithm Index	ABC	GSO	ACS	PSO	GA	QGA	Ant-Q	TBO
Convergence time (s)	72.3	63.4	228	102	48.3	47.8	115	3.37
P_{loss} (MW)	3.82×10^4	3.86×10^4	3.83×10^4	3.81×10^4	3.77×10^4	3.76×10^4	3.74×10^4	3.75×10^4
V_{d} (%)	8.34×10^3	8.87×10^3	7.36×10^3	8.07×10^3	7.78×10^3	7.56×10^3	7.14×10^3	6.94×10^3
f	2.33×10^4	2.38×10^4	2.28×10^4	2.31×10^4	2.28×10^4	2.26×10^4	2.23×10^4	2.22×10^4
Best	2.32×10^4	2.37×10^4	2.28×10^4	2.30×10^4	2.27×10^4	2.26×10^4	2.23×10^4	2.22×10^4
Worst	2.33×10^4	2.38×10^4	2.28×10^4	2.32×10^4	2.28×10^4	2.26×10^4	2.23×10^4	2.22×10^4
Variance	381	1.29×10^3	228	2.37×10^3	178	194	221	66.4
Std. Dev.	19.5	36	15.1	48.7	13.4	13.9	14.9	8.15
Rel. Std. Dev	8.39×10^{-4}	1.51×10^{-3}	6.61×10^{-4}	2.11×10^{-3}	5.87×10^{-4}	6.16×10^{-4}	6.67×10^{-4}	3.67×10^{-4}

On the other hand, Table 3 shows that the convergence stability of the TBO was the highest as the values of all of its indices were the lowest, and Rel. Std. Dev. of the TBO was only 17.39% with respect to that of PSO gotten from the IEEE 300-bus system and was up to 75.79% of that of ABC gotten from the IEEE 118-bus system. This was due to the exploitation of prior knowledge by scouts and workers, which beneficially avoids the randomness of global search, thus a higher search efficiency can be achieved.

5. Conclusions

In this article, a novel TBO incorporating behavior conversion was obtained for RPO. Like other AI algorithms, the TBO is highly independent from the accurate system model and has a much stronger global search ability and a higher application flexibility compared with the traditional mathematical optimization methods. Compared with network simplified model (e.g., external equivalent model)

-based methods, the TBO also can rapidly converge to an optimum for RPO but it can obtain a higher quality optimum via global optimization. By introducing the Q-learning-based optimization, the TBO can learn, store, and transfer knowledge between different optimization tasks, in which the state-action chain can significantly reduce the size of the Q-value matrix, and the cooperative exploration between scouts and workers can dramatically accelerate knowledge learning. Compared with the general AI algorithms, the greatest advantage of the TBO is that it can significantly accelerate the convergence rate for a new task via re-using prior knowledge from the source tasks. Through simulation comparisons on the IEEE 118-bus system and IEEE 300-bus system, the convergence rate of the TBO was 4 to 68 times faster than that of existing AI algorithms for RPO, while ensuring the quality and convergence stability of the optimal solutions. Thanks to its superior optimization performance, the TBO can be readily applied to other cases of nonlinear programming of large-scale power systems.

Author Contributions: Preparation of the manuscript was performed by H.C., T.Y., X.Z., B.Y., and Y.W.

Funding: This research was funded by [National Natural Science Foundation of China] grant number [51777078], and [Yunnan Provincial Basic Research Project-Youth Researcher Program] grant number [2018FD036], and The APC was funded by [National Natural Science Foundation of China].

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

References

1. Grudinin, N. Reactive power optimization using successive quadratic programming method. *IEEE Trans. Power Syst.* **1998**, *13*, 1219–1225. [CrossRef]
2. Li, C.; Johnson, R.B.; Svoboda, A.J. A new unit commitment method. *IEEE Trans. Power Syst.* **1997**, *12*, 113–119.
3. Zhou, B.; Chan, K.W.; Yu, T.; Chung, C.Y. Equilibrium-inspired multiple group search optimization with synergistic learning for multiobjective electric power dispatch. *IEEE Trans. Power Syst.* **2013**, *28*, 3534–3545. [CrossRef]
4. Sun, D.I.; Ashley, B.; Brewer, B. Optimal power flow by newton approach. *IEEE Trans. Power Appar. Syst.* **1984**, *103*, 2864–2880. [CrossRef]
5. Fan, J.Y.; Zhang, L. Real-time economic dispatch with line flow and emission constraints using quadratic programming. *IEEE Trans. Power Syst.* **1998**, *13*, 320–325. [CrossRef]
6. Wei, H.; Sasaki, H.; Kubokawa, J.; Yokoyama, R. An interior point nonlinear programming for optimal power flow problems with a novel data structure. *IEEE Trans. Power Syst.* **1998**, *13*, 870–877. [CrossRef]
7. Dai, C.; Chen, W.; Zhu, Y.; Zhang, X. Seeker optimization algorithm for optimal reactive power dispatch. *IEEE Trans. Power Syst.* **2009**, *24*, 1218–1231.
8. Abu-Mouti, F.S.; El-Hawary, M.E. Optimal distributed generation allocation and sizing in distribution systems via artificial bee colony algorithm. *IEEE Trans. Power Del.* **2011**, *26*, 2090–2101. [CrossRef]
9. Han, Z.; Zhang, Q.; Shi, H.; Zhang, J. An improved compact genetic algorithm for scheduling problems in a flexible flow shop with a multi-queue buffer. *Processes* **2019**, *7*, 302. [CrossRef]
10. Han, P.; Fan, G.; Sun, W.; Shi, B.; Zhang, X. Research on identification of LVRT characteristics of photovoltaic inverters based on data testing and PSO algorithm. *Processes* **2019**, *7*, 250. [CrossRef]
11. He, S.; Wu, Q.H.; Saunders, J.R. Group search optimizer: An optimization algorithm inspired by animal searching behavior. *IEEE Trans. Evol. Comput.* **2009**, *13*, 973–990. [CrossRef]
12. Gómez, J.F.; Khodr, H.M.; De Oliveira, P.M.; Ocque, L.; Yusta, J.M.; Urdaneta, A.J. Ant colony system algorithm for the planning of primary distribution circuits. *IEEE Trans. Power Syst.* **2004**, *19*, 996–1004. [CrossRef]
13. Yang, B.; Yu, T.; Zhang, X.S.; Huang, L.N.; Shu, H.C.; Jiang, L. Interactive teaching–learning optimiser for parameter tuning of VSC-HVDC systems with offshore wind farm integration. *IET Gener. Transm. Distrib.* **2017**, *12*, 678–687. [CrossRef]
14. Yang, B.; Zhang, X.S.; Yu, T.; Shu, H.C.; Fang, Z.H. Grouped grey wolf optimizer for maximum power point tracking of doubly-fed induction generator based wind turbine. *Energy Convers. Manag.* **2017**, *133*, 427–443. [CrossRef]

15. Yang, B.; Zhong, L.E.; Zhang, X.S.; Shu, H.C.; Li, H.F.; Jiang, L.; Sun, L.M. Novel bio-inspired memetic salp swarm algorithm and application to MPPT for PV systems considering partial shading condition. *J. Clean. Prod.* **2019**, *215*, 1203–1222. [CrossRef]
16. Yang, B.; Yu, T.; Zhang, X.S.; Li, H.F.; Shu, H.C.; Sang, Y.Y.; Jiang, L. Dynamic leader based collective intelligence for maximum power point tracking of PV systems affected by partial shading condition. *Energy Convers. Manag.* **2019**, *179*, 286–303. [CrossRef]
17. Montoya, F.G.; Alcayde, A.; Arrabal-Campos, F.M.; Raul, B. Quadrature current compensation in non-sinusoidal circuits using geometric algebra and evolutionary algorithms. *Energies* **2019**, *12*, 692. [CrossRef]
18. Yu, T.; Liu, J.; Chan, K.W.; Wang, J.J. Distributed multi-step Q (λ) learning for optimal power flow of large-scale power grids. *Int. J. Electr. Power Energy Syst.* **2012**, *42*, 614–620. [CrossRef]
19. Mühürcü, A. FFANN optimization by ABC for controlling a 2nd order SISO system's output with a desired settling time. *Processes* **2019**, *7*, 4. [CrossRef]
20. Sundareswaran, K.; Sankar, P.; Nayak, P.S.R.; Simon, S.P.; Palani, S. Enhanced energy output from a PV system under partial shaded conditions through artificial bee colony. *IEEE Trans. Sustain. Energy* **2014**, *6*, 198–209. [CrossRef]
21. Chandrasekaran, K.; Simon, S.P. Multi-objective unit commitment problem with reliability function using fuzzified binary real coded artificial bee colony algorithm. *IET Gener. Transm. Distrib.* **2012**, *6*, 1060–1073. [CrossRef]
22. Gao, Z. Advances in Modelling, monitoring, and control for complex industrial systems. *Complexity* **2019**, *2019*, 2975083. [CrossRef]
23. Tognete, A.L.; Nepomuceno, L.; Dos Santos, A. Framework for analysis and representation of external systems for online reactive-optimisation studies. *IEE Gener. Transm. Distrib.* **2005**, *152*, 755–762. [CrossRef]
24. Tognete, A.L.; Nepomuceno, L.; Dos Santos, A. Evaluation of economic impact of external equivalent models used in reactive OPF studies for interconnected systems. *IET Gener. Transm. Distrib.* **2007**, *1*, 140–145. [CrossRef]
25. Britannica Academic, S.V. The Life of Bee. Available online: http://academic.eb.com/EBchecked/topic/340282/The-Life-of-the-Bee (accessed on 24 April 2019).
26. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
27. Ramon, J.; Driessens, K.; Croonenborghs, T. Transfer learning in reinforcement learning problems through partial policy recycling. In *Learn ECML*; Springer: Heidelberg/Berlin, Germany, 2007; pp. 699–707.
28. Gao, Z.; Saxen, H.; Gao, C. Data-driven approaches for complex industrial systems. *IEEE Trans. Ind. Inform.* **2013**, *9*, 2210–2212. [CrossRef]
29. Taylor, M.E.; Stone, P. Transfer learning for reinforcement learning domains: A survey. *J. Mach. Learn. Res.* **2009**, *10*, 1633–1685.
30. Pan, J.; Wang, X.; Cheng, Y.; Cao, G. Multi-source transfer ELM-based Q learning. *Neurocomputing* **2014**, *137*, 57–64. [CrossRef]
31. Bianchi, R.A.C.; Celiberto, L.A.; Santos, P.E.; Matsuura, J.P.; Mantaras, R.L.D. Transferring knowledge as heuristics in reinforcement learning: A case-based approach. *Artif. Intell.* **2015**, *226*, 102–121. [CrossRef]
32. Watkins, J.C.H.; Dayan, P. *Q*-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
33. Ghavamzadeh, M.; Mahadevan, S. Hierarchical average reward reinforcement learning. *J. Mach. Learn. Res.* **2007**, *8*, 2629–2669.
34. Dietterich, T.G. Hierarchical reinforcement learning with the MAXQ value function decomposition. *J. Artif. Intel.* **2000**, *13*, 227–303. [CrossRef]
35. Yu, T.; Zhou, B.; Chan, K.W.; Chen, L.; Yang, B. Stochastic optimal relaxed automatic generation control in Non-Markov environment based on multi-step Q (λ) learning. *IEEE Trans. Power Syst.* **2011**, *26*, 1272–1282. [CrossRef]
36. Maloosini, A.; Blanzieri, E.; Calarco, T. Quantum genetic optimization. *IEEE Trans. Evol. Comput.* **2008**, *12*, 231–241. [CrossRef]
37. Dorigo, M.; Gambardella, L.M. A study of some properties of Ant-Q. In *International Conference on Parallel Problem Solving from Nature*; Springer-Verlag: Berlin, Germany, 1996; pp. 656–665.
38. Zhang, X.S.; Yu, T.; Yang, B.; Cheng, L. Accelerating bio-inspired optimizer with transfer reinforcement learning for reactive power optimization. *Knowl. Base. Syst.* **2017**, *116*, 26–38. [CrossRef]

39. Rajaraman, P.; Sundaravaradan, N.A.; Mallikarjuna, B.; Jaya, B.R.M.; Mohanta, D.K. Robust fault analysis in transmission lines using Synchrophasor measurements. *Prot. Control. Mod. Power Syst.* **2018**, *3*, 108–110.
40. Hou, K.; Shao, G.; Wang, H.; Zheng, L.; Zhang, Q.; Wu, S.; Hu, W. Research on practical power system stability analysis algorithm based on modified SVM. *Prot. Control Mod. Power Syst.* **2018**, *3*, 119–125. [CrossRef]
41. Ren, C.; Xu, Y.; Zhang, Y.C. Post-disturbance transient stability assessment of power systems towards optimal accuracy-speed tradeoff. *Prot. Control Mod. Power Syst.* **2018**, *3*, 194–203. [CrossRef]
42. Dabra, V.; Paliwal, K.K.; Sharma, P.; Kumar, N. Optimization of photovoltaic power system: A comparative study. *Prot. Control Mod. Power Syst.* **2017**, *2*, 29–39. [CrossRef]
43. Yang, B.; Yu, T.; Shu, H.C.; Dong, J.; Jiang, L. Robust sliding-mode control of wind energy conversion systems for optimal power extraction via nonlinear perturbation observers. *Appl. Energy* **2018**, *210*, 711–723. [CrossRef]
44. Yang, B.; Jiang, L.; Wang, L.; Yao, W.; Wu, Q.H. Nonlinear maximum power point tracking control and modal analysis of DFIG based wind turbine. *Int. J. Electr. Power Energy Syst.* **2016**, *74*, 429–436. [CrossRef]