Article

Quantum-Inspired Differential Evolution with Grey Wolf Optimizer for 0-1 Knapsack Problem

Yule Wang and Wanliang Wang *

College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China; 1111712014@zjut.edu.cn
* Correspondence: zjutwwl@zjut.edu.cn

Abstract: The knapsack problem is one of the most widely researched NP-complete combinatorial optimization problems and has numerous practical applications. This paper proposes a quantum-inspired differential evolution algorithm with grey wolf optimizer (QDGWO) to enhance the diversity and convergence performance and improve the performance in high-dimensional cases for 0-1 knapsack problems. The proposed algorithm adopts quantum computing principles such as quantum superposition states and quantum gates. It also uses adaptive mutation operations of differential evolution, crossover operations of differential evolution, and quantum observation to generate new solutions as trial individuals. Selection operations are used to determine the better solutions between the stored individuals and the trial individuals created by mutation and crossover operations. In the event that the trial individuals are worse than the current individuals, the adaptive grey wolf optimizer and quantum rotation gate are used to preserve the diversity of the population as well as speed up the search for the global optimal solution. The experimental results for 0-1 knapsack problems confirm the advantages of QDGWO with the effectiveness and global search capability for knapsack problems, especially for high-dimensional situations.

Keywords: quantum computing; differential evolution; grey wolf optimizer; evolutionary algorithm; 0-1 knapsack problem

check for updates

Citation: Wang, Y.; Wang, W. Quantum-Inspired Differential Evolution with Grey Wolf Optimizer for 0-1 Knapsack Problem. *Mathematics* **2021**, *9*, 1233. https://doi.org/10.3390/math9111233

Academic Editor: Amir H. Alavi

Received: 1 March 2021
Accepted: 5 May 2021
Published: 28 May 2021

1. Introduction

The 0-1 knapsack problem (KP01) is a classical combinatorial optimization problem. It has many practical applications, such as project selection, investment decisions, and complexity theory [1,2]. Two classes of approaches were previously proposed to solve the KP01 [3]. The first class of approaches includes exact methods based on mathematical programming and operational research. It is possible to obtain the exact solutions of small-scale KP01 problems by exact methods such as branching and bound algorithm [4] and dynamic programming [5]. However, KP01 problems in various complex situations are NP-hard problems, and it is impractical to obtain optimal solutions using deterministic optimization methods for large-scale problems. The second class contains approximate methods based on metaheuristic algorithms [6]. Metaheuristic algorithms are shown to be effective approaches to solving complex engineering problems in a reasonable time when compared with exact methods [7]. Therefore, the application of metaheuristic algorithms has drawn a great deal of attention in the field of optimization.

In recent years, most of the metaheuristic algorithms, such as the genetic algorithm (GA) [8], ant colony optimization (ACO) [9], particle swarm algorithm (PSO) [10], artificial bee colony (ABC) [11], cuckoo search (CS) [12], firefly algorithm (FA) [13], and improved approaches based on these algorithms [14–19], were applied to KP01 problems and achieved outstanding results. However, these metaheuristic algorithms require not only a large amount of memory for storing the population of solutions, but also a long computational time for finding the optimal solutions. In the last few years, novel metaheuristic algorithms were proposed. Wang et al. [20,21] improved the swarm intelligence optimization approach inspired by the herding behavior of krill and came up with the krill herd (KH) algorithm to solve combinatorial optimization problems. Faramarzi et al. [22] presented a metaheuristic called the Marine Predators Algorithm (MPA) with an application in engineering design problems. MPA follows the rules that naturally govern in optimal foraging strategy and encounters a rate policy between predator and prey in marine ecosystems. Inspired by the phototaxis and Lévy flights of moths, Wang et al. [23] developed a new metaheuristic algorithm called the moth search (MS) algorithm. MS was applied to solve discounted 0-1 knapsack problems [24] and set-union knapsack problems [25]. Gao et al. [26] presented a novel selection mechanism augmenting the generic DE algorithm (NSODE) to achieve better optimization results for solving fuzzy job-shop scheduling problems. Abualigah et al. [27] proposed the arithmetic optimization algorithm (AOA) based on the distribution behavior of the main arithmetic operators in mathematics: multiplication, division, subtraction, and addition. Wang et al. [28] proposed a new nature-inspired metaheuristic algorithm called monarch butterfly optimization (MBO) by simulating the migration of monarch butterflies. MBO was applied to solve classic KP01 [29], discounted KP01 [30], and large-scale KP01 [31,32] problems with superior searching accuracy, convergent capability, and stability. In most metaheuristic algorithms, it is difficult to use the information from individuals in previous iterations in the updating process. Wang et al. [33] presented a method for reusing the information available from previous individuals and feeding previous useful information back into the updating process in order to guide later searches.

The emergence of quantum computing [34,35] was derived from the principles of quantum theory such as quantum superposition, quantum entanglement, quantum interference, and quantum collapse. Quantum computing brings new ideas to optimization due to its underlying concepts, along with the ability to process huge numbers of quantum states simultaneously in parallel. The merging of metaheuristic optimization and quantum computing recently became a growing theoretical and practical interest aiming at deriving benefits from quantum computing capabilities to enhance the convergence and speed of metaheuristic algorithms. Several scholars investigated the effect of introducing quantum computing in metaheuristic algorithms to maintain a balance between exploration and exploitation. Han and Kim proposed a genetic quantum algorithm (GQA) [36] and a quantum inspired evolutionary algorithm (QEA) [37] by merging classical evolutionary algorithms with quantum computing concepts such as the quantum bit and the quantum rotation gate. Talbi et al. [38] proposed a new algorithm inspired by genetic algorithms and quantum computing for solving the traveling salesman problem (TSP). Chang et al. [39] proposed a quantum-inspired electromagnetism-like mechanism (QEM) to solve the KP01. Xiong et al. [40] presented an analysis of quantum rotation gates in quantum-inspired evolutionary algorithms. To avoid the problem of premature convergence, mutation operation [41], crossover operation [42], new termination criterion, and new rotation gate [43] were applied to the QEA and subsequently improved.

The differential evolution (DE) algorithm [44], proposed by Storn and Price, was derived from differential vectors of solutions for global optimization. Several simple operations including mutation, crossover, and selection were used in the DE algorithm to explore the search space. Subsequently, several algorithms combining the DE algorithm with quantum computing were designed to increase global search ability. Hota and Pat [45] extended the concept of differential operators with adaptive parameter control to the quantum paradigm and proposed the adaptive quantum-inspired differential evolution (AQDE) algorithm. In addition, quantum interference operation [46] and mutation operation [47] were brought into a quantum-inspired DE algorithm.

Several metaheuristic algorithms combining the QEA and DE were proven to be effective and efficient for solving the KP01. Wang et al. [48] proposed a quantum swarm evolutionary (QSE) algorithm that updated quantum angles automatically with improved PSO. Layeb [49] presented a quantum inspired harmony search algorithm (QIHSA) based on a harmony search algorithm (HSA) and quantum computing. Zouache et al. [50] proposed a merged algorithm called quantum-inspired differential evolution with particle swarm optimization (QDEPSO) to solve the KP01. Gao et al. [51] proposed a quantum-inspired wolf pack algorithm (QWPA) with quantum rotation and quantum collapse to improve the performance of the wolf pack algorithm for the KP01.

The grey wolf optimizer (GWO) proposed by Mirjalili et al. [52] mimics the specific behavior of grey wolves based on leadership hierarchy in nature. Srikanth et al. [53] presented a quantum-inspired binary grey wolf optimizer (QIBGWO) to solve the problem of unit commitment scheduling.

Population diversity is crucial in evolutionary algorithms to enable global exploration and to avoid poor performance due to premature convergence [54]. However, it is hard for classical algorithms to enhance diversity and convergence performance because their population quickly converges to a specific region of the solution space. In addition, these algorithms require a large amount of memory as well as long computational time to find the optimal solution for high-dimensional situations. To avoid these difficulties, we propose a new algorithm, called quantum-inspired differential evolution algorithm with grey wolf optimizer (QDGWO). The proposed algorithm combines the features of the QEA, DE, and GWO to solve the 0-1 knapsack problem. To preserve diversity throughout the evolution, the new algorithm adopts the concepts of quantum representation and the integration of the quantum operators such as quantum measurement and quantum rotation. The adaptive operations of the DE (mutation, crossover, selection) and GWO can increase the adaptation and diversification in updating individuals. The experimental results demonstrate the competitive performance of the proposed algorithm.

The rest of the paper is organized as follows: Section 2 defines the 0-1 knapsack problem. The proposed QDGWO algorithm is presented in Section 3. The experimental results and discussion are summarized in Section 4. Conclusions and directions for future work are discussed in Section 5.

2. Related Work

2.1. Knapsack Problem

The 0-1 knapsack problem is a well-known combinatorial optimal problem that has been studied in areas such as project selection, resource distribution, and the network interdiction problem. The KP01 was demonstrated to be NP-complete [55,56]. It can be described as follows:

Given a set W of m items, $W = (x_1, x_2, x_3, \ldots, x_m)$. w_i is the weight item x_i, and p_i is the profit of x_i. C is the weight capacity of the knapsack. The objective is to find the subset X_{optimal} from set of m items that maximizes the total profit, while keeping the total weight of the selected items from exceeding C.

The 0-1 knapsack problem can be defined as:

$$\begin{aligned} &\text{Maximize } f(x) = \sum_{i=1}^{m} p_i x_i \\ &\text{s.t. } \sum_{i=1}^{m} w_i x_i \leq C, x_i \in \{0,1\}, \forall i \in \{1,2,\ldots,m\} \end{aligned} \tag{1}$$

where x_i can take either the value 1 (as selected) or the value 0 (as not selected, also called rejected).

2.2. Grey Wolf Optimizer (GWO)

The GWO algorithm [52] is inspired by the leadership hierarchy and hunting mechanism of grey wolves. To model the social order of grey wolves in the GWO, the best solution is considered the alpha (α) wolf, and the second and third best solutions are beta (β) and delta (δ) wolves, respectively. The rest of the feasible solutions are considered as omega (ω) wolves. In the GWO, α, β, and δ wolves lead the hunting, and the ω wolves go after these leading wolves when searching for the global optimal solution (target) as the prey, as shown in Figure 1.

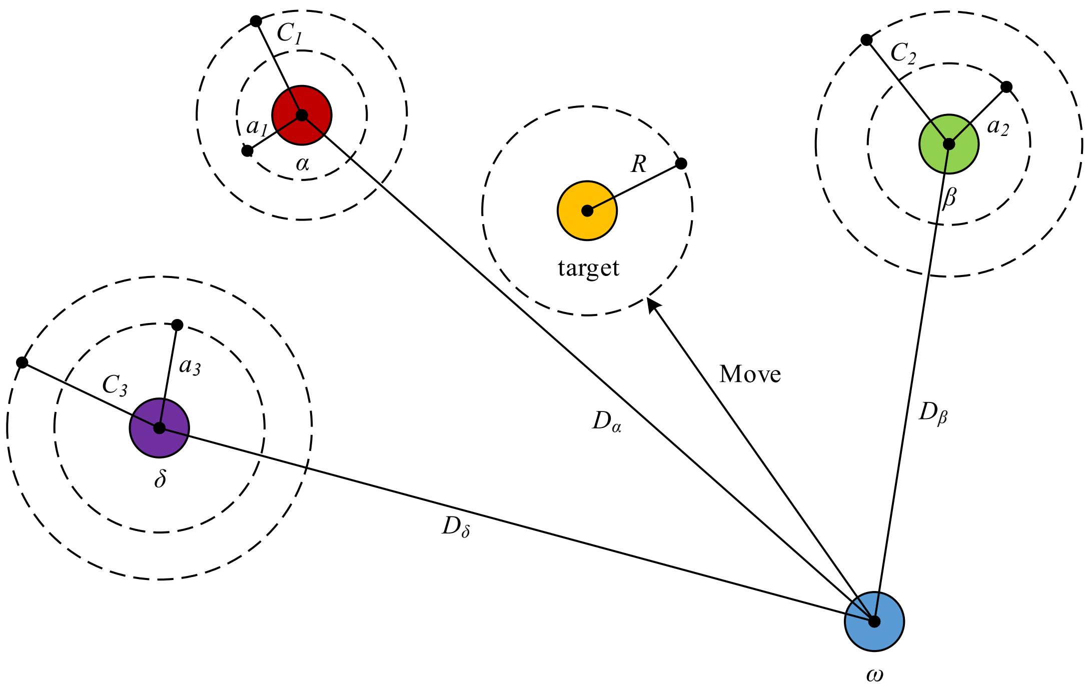

Figure 1. Position updating mechanism of GWO.

Grey wolves have the ability to recognize the location of prey and encircle them during the hunt. In order to simulate the hunting behavior of grey wolves mathematically, it is supposed that the α, β, and δ wolves have better knowledge about the potential location of prey. Therefore, the GWO saves the first three best solutions arrived at so far and forces the other ω wolves to update their positions according to the position of the best search agents.

During optimization, the GWO algorithm allows its search agents to update their position based on the location of the alpha, beta, and delta wolves with the distance vector between itself and the three best wolves when attacking the prey. Finally, the position and fitness of the alpha wolf are regarded as the global optimal solution in searching for the optimization when a termination criterion is satisfied.

3. Quantum-Inspired Differential Evolution with Adaptive Grey Wolf Optimizer

The proposed QDGWO algorithm is presented for the knapsack problems. First, the proposed algorithm adopts the quantum computing principles such as quantum representation and quantum measurement operation. Quantum representation allows the representation of the superposition of all potential states in one quantum individual. Second, adaptive mutation operations (used in the DE), crossover operations (used in the DE), and quantum observation are combined to generate new solutions as trial individuals in the solution space. Finally, the selection operator chooses the better solutions between the stored individuals and the trial individuals generated by the mutation and crossover operations of the DE. In the event that the trial individuals are worse than the current individuals, the QDGWO integrates the adaptive GWO and quantum rotation gate to preserve the diversity of the population of solutions as well as accelerate the search for the global optimum. The framework of the QDGWO algorithm is shown in Figure 2.

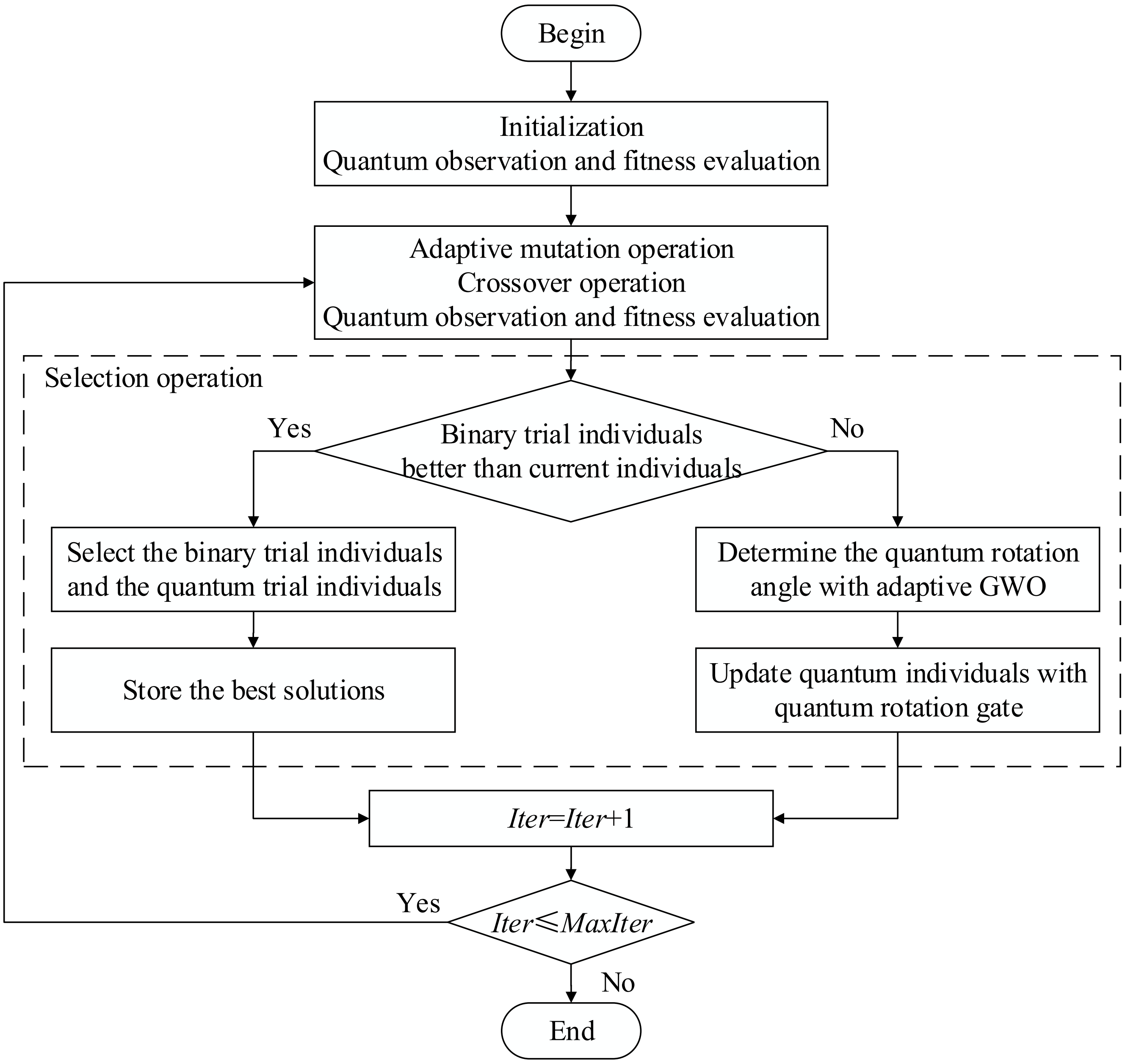
Begin
Initialization
Quantum observation and fitness evaluation
Adaptive mutation operation
Crossover operation
Quantum observation and fitness evaluation
Selection operation
Binary trial individuals better than current individuals
Yes
No
Select the binary trial individuals and the quantum trial individuals
Store the best solutions
Determine the quantum rotation angle with adaptive GWO
Update quantum individuals with quantum rotation gate
Iter=Iter+1
Yes
Iter≤MaxIter
No
End

Figure 2. Framework of QDGWO.

3.1. Binary Representation

The choice of the representation for individuals, also known as individual coding, is a crucial issue in evolutionary algorithms. The proposed QDGWO algorithm adopts the binary coding, which is the most appropriate way to indicate the selection or rejection of items. Each individual X is represented as a binary vector with length m (m is the item size): $X = (x_1, x_2, \ldots, x_m)$, where m is the number of items.

$$\begin{cases} x_i = 1, & \text{if item } x_i \text{ is selected} \\ x_i = 0, & \text{if item } x_i \text{ is rejected} \end{cases} \tag{2}$$

The following example shows the binary representation for item selection: x_1 and x_3 from the item set W are selected: $W = \{x_1, x_2, x_3, x_4\} \rightarrow X = (1\ 0\ 1\ 0)$.

The binary population $P(t) = \left(X_1^t, X_2^t, \ldots, X_n^t\right)$ is made up of the binary individuals at the tth generation, where n is the size of population.

3.2. Quantum Representation

The representation of the AQDE [45] is used in the proposed algorithm, where each quantum individual q corresponds to a phase vector q_θ, which is a string of phase angles θ_i ($1 \le i \le m$), which can be given by

$$q_\theta = [\theta_1, \theta_2, \ldots, \theta_m], \theta_i \in [0, 2\pi] \tag{3}$$

where m is the length of the quantum bit (qubit) individual.

Each quantum individual q is a string of qubits:

$$q = \left[\begin{array}{c|c|c|c} \cos(\theta_1) & \cos(\theta_2) & \multirow{2}{*}{\ldots} & \cos(\theta_m) \\ \sin(\theta_1) & \sin(\theta_2) & & \sin(\theta_m) \end{array} \right] \tag{4}$$

The probability amplitudes of a quantum bit are expressed as a pair of numbers ($\cos(\theta_i)$, $\sin(\theta_i)$). $|\sin(\theta_i)|^2$ represents the probability of selecting item x_i, and $|\cos(\theta_i)|^2$ represents the probability of rejecting item x_i.

The quantum population $Q(t) = \left(q_1^t, q_2^t, \ldots, q_n^t\right)$ is made up of the quantum individuals at the tth generation, where n is the size of the population.

3.3. Initialization

The general principle of superposition of quantum mechanics assumes that the original state must be considered as the result of a superposition of two or more other states in an infinite number of ways [57]. Therefore, the initial quantum individual is regarded as a superposition of all possible states. For 0-1 knapsack problems, each state represents a combination of selecting or rejecting items, and the initial quantum individual is required to generate every possible combination. For this, each vector $q_{\theta i}$ is initialized by:

$$q_{\theta i}^{t=0} = \left(\left(\frac{\pi}{4}\right) \cdot r_{i1}, \ldots, \left(\frac{\pi}{4}\right) \cdot r_{im} \right) \tag{5}$$

where r_{ij} = random{1,3,5,7}, which means r_{ij} is an odd integer generated randomly in the set $r_{ij} \in \{1,3,5,7\}$ while $\theta_{ij} \in \left\{\frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}\right\}$. This means that for initial individuals, $|\cos(\theta_{ij})|^2 = |\sin(\theta_{ij})|^2 = 1/2$, so that the probabilities of selecting item x_i and rejecting item x_i are equal.

The initial quantum individual $q_i^{t=0}$ which corresponds to $q_{\theta i}^{t=0}$ can be given by:

$$q_i^{t=0} = \left[\begin{array}{c|c|c|c} \cos\theta_{i1}^0 & \cos\theta_{i2}^0 & \multirow{2}{*}{\ldots} & \cos\theta_{im}^0 \\ \sin\theta_{i1}^0 & \sin\theta_{i2}^0 & & \sin\theta_{im}^0 \end{array} \right] \tag{6}$$

where m is the length of the qubit quantum individual.

$Q(0) = \left(q_1^0, q_2^0, \ldots, q_n^0\right)$ is the initial quantum population, where n is the size of the population.

3.4. Quantum Observation and Fitness Evaluation

Based on the quantum superposition principle, a quantum state is a superposition of all possibly stationary states. The quantum superposition state will collapse to a stationary state by quantum observation. In the proposed QDGWO algorithm, the quantum superposition states are represented by quantum individuals, and the stationary states are represented through binary individuals. Before evaluating the fitness of individuals, quantum observation and reparation for quantum individuals q are required to receive binary individuals X, as shown in Algorithm 1.

After quantum observation, the fitness of binary individual X is evaluated as:

$$f\left(X_i^t\right) = \sum_{i=1}^{m} p_i x_i^t \tag{7}$$

Algorithm 1 Quantum Observation and Reparation

```
Input: quantum individuals $q$
Output: binary individuals $X$
$x_i \leftarrow 0$ //Initialize the bits of individual $X$ to 0.
$w_{\text{total}} \leftarrow 0$ // Initialize the total weights of the individuals to 0.
while ($totalw \leq C$) do
    $i \leftarrow$ rand_i$[1, m]$ //Generate the random integer $i \in \{1, 2, \ldots, m\}$.
    if ($x_i = 0$) then $r \leftarrow \text{rand}(0, 1)$
        if($r > |\cos(\theta_i)|^2$) then
            $x_i \leftarrow 1$
            $w_{\text{total}} \leftarrow w_{\text{total}} + w_i$ //Select item $x_i$ and include the weight $w_i$ of item $x_i$ in the total weight $w_{total}$.
        end if
    end if
end while
$x_i \leftarrow 0$
$w_{\text{total}} \leftarrow w_{\text{total}} - w_i$ //The total weight $w_{total}$ has exceeded the capacity $C$ when the loop is ended, so item $x_i$ needs to be extracted from the selected items for reparation.
```

3.5. Adaptive Mutation Operation with Dynamic Iteration Factor

The mutation operation is one of the main operations in differential evolution. In the QDGWO, DE/best/2, proposed by Price and Storn [44], is used to select parent vectors. In this strategy, the mutation vector $q_{\theta i}^{\mathrm{M}t}$ is generated by the vector $q_{\theta\alpha}$ corresponding to the current best binary individual X_α and the difference between two different target vectors $q_{\theta r1}$ and $q_{\theta r2}$ which are randomly selected. This difference is weighted by the differentiation control factor F.

The quantum mutation vector $q_{\theta i}^{\mathrm{M}t}$ at the tth generation can be generated by:

$$q_{\theta i}^{\mathrm{M}t} = q_{\theta\alpha} + F^t(q_{\theta r1} - q_{\theta r2}) \tag{8}$$

where $r_1 \in \{1, 2, \ldots, m\}$ and $r_1 \neq i; r_2 \in \{1, 2, \ldots, m\}$ and $r_2 \neq i; r_1 \neq r_2$.

To improve the performance of differential operations in different phases, we propose an adaptive strategy to determine the differentiation control factor F with the iteration of evolution:

$$F^t = F_0 + F_1 \cdot 2^{\omega} \cdot \text{rand}(0, 1) \tag{9}$$

$$\omega = e^{1 - \frac{t_{\max}}{t_{\max} - t}} \tag{10}$$

where F_0 is the initial differentiation control factor, and F_1 is the adaptive f differentiation control factor. $t_{\max}$ is the maximum iteration number of the algorithm.

With this adaptive strategy, in the early stage of the iteration, the smaller t will be proposed with a larger F^t, which is beneficial for attaining good diversity of individuals for global searching. F^t will be smaller and smaller during the iteration. In the late stages, F^t is close to F_0, which aids in local searching for the global optimal solution.

The quantum mutation individual $q_i^{\mathrm{M}t}$ corresponding to the quantum mutant vector $q_{\theta i}^{\mathrm{M}t}$ can be obtained by:

$$q_i^{\mathrm{M}t} = \left[\begin{array}{c|c|c|c} \cos\theta_{i1}^{\mathrm{M}t} & \cos\theta_{i2}^{\mathrm{M}t} & \multirow{2}{*}{$\cdots$} & \cos\theta_{im}^{\mathrm{M}t} \\ \sin\theta_{i1}^{\mathrm{M}t} & \sin\theta_{i2}^{\mathrm{M}t} & & \sin\theta_{im}^{\mathrm{M}t} \end{array}\right] \tag{11}$$

where m is the length of the qubit quantum individual.

The quantum mutation population $Q^{\mathrm{M}}(t) = (q_1^{\mathrm{M}t}, q_2^{\mathrm{M}t}, \ldots, q_n^{\mathrm{M}t})$ is made up of the quantum mutation individuals at the tth generation, where n is the size of the population.

3.6. Crossover Operation

The crossover operation is another main operation in differential evolution. The trial vector $q_{\theta i}^{Ct}$ is generated by crossover between the mutant vector $q_{\theta i}^{Mt}$ and the target vector $q_{\theta i}^{t}$ with a binomial crossover strategy [44].

The quantum trial vector $q_{\theta i}^{Ct}$ at the tth generation can be generated by:

$$q_{\theta ij}^{Ct} = \begin{cases} q_{\theta ij}^{Mt}, \text{ if } \left(rand_j(0,1) \le CR^t\right) \text{ or } (j = rnbr\_i) \\ q_{\theta ij}^{t}, \text{ if } \left(rand_j(0,1) > CR^t\right) \text{ and } (j \neq rnbr\_i) \end{cases} \tag{12}$$

where $CR \in [0,1]$ is the probability of the crossover operation which is randomly generated at tth iteration. In addition, $rnbr\_i \in \{1, m\}$ is an integer to ensure that $q_{\theta i}^{Ct}$ obtains at least one vector from $q_{\theta i}^{Mt}$.

The quantum trial individual q_i^{Ct} corresponding to the quantum trial vector $q_{\theta i}^{Ct}$ can be obtained by:

$$q_i^{Ct} = \left[\begin{array}{c|c|c|c} \cos\theta_{i1}^{Ct} & \cos\theta_{i2}^{Ct} & \multirow{2}{*}{\ldots} & \cos\theta_{im}^{Ct} \\ \sin\theta_{i1}^{Ct} & \sin\theta_{i2}^{Ct} & & \sin\theta_{im}^{Ct} \end{array} \right] \tag{13}$$

where m is the length of the qubit quantum individual.

The quantum trial population $Q^C(t) = \left(q_1^{Ct}, q_2^{Ct}, \ldots, q_n^{Ct}\right)$ is made up of the quantum trial individuals at the tth generation, where n is the size of the population.

3.7. Selection Operation

After the crossover operation, the trial quantum individuals will be transformed into binary individuals by observation and reparation, as discussed in Section 3.4. The population of trial quantum individuals $Q^C(t)$ is transformed into a population of trial binary individuals $P^C(t) = \left\{X_1^{Ct}, X_2^{Ct}, \ldots, X_n^{Ct}\right\}$.

The selection operation generates the individuals of next iteration X_i^{t+1} between the current individuals X_i^t and the trial binary individuals X_i^{Ct}, as follows:

$$X_i^{t+1} = \begin{cases} X_i^{Ct}, \text{ if } \left(f(X_i^{Ct}) > f(X_i^t)\right) \\ X_i^t, \text{ if } \left(f(X_i^{Ct}) \le f(X_i^t)\right) \end{cases} \tag{14}$$

The quantum individuals of the next iteration are generated by:

$$q_{\theta i}^{t+1} = \begin{cases} q_{\theta i}^{Ct}, \text{ if } \left(f(X_i^{Ct}) > f(X_i^t)\right) \\ \text{update } q_{\theta i}^{t} \text{ by } R_{\text{GWO}}, \text{ if } \left(f\left(X_i^{Ct}\right) \le f\left(X_i^t\right)\right) \end{cases} \tag{15}$$

where R_{GWO} is a quantum rotation gate (QRG) with an adaptive GWO. R_{GWO} is presented in Section 3.8.

3.8. Quantum Rotation Gate with Adaptive GWO

The quantum rotation gate $U(\theta_i)$ is used to update the values of the qubits in a quantum individual as follows [36]:

$$U(\theta_i) = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \tag{16}$$

The quantum individuals of the next iteration after quantum rotation are presented as:

$$\begin{bmatrix} \alpha'_i \\ \beta'_i \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} \tag{17}$$

where $\theta_i = s(\alpha_i \beta_i)\Delta\theta_i$ is the rotation angle of the QRG, and $s(\alpha_i\ \beta_i)$ is the direction signal of the rotation angle.

The polar plot of the QRG for qubits is illustrated in Figure 3, and the quantum rotation angle parameters used in [36] are shown in Table 1.

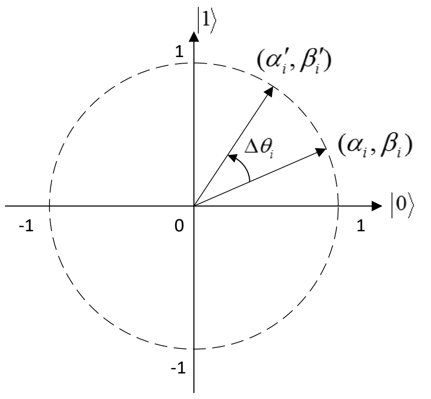

Figure 3. Polar plot of quantum rotation gate for qubit.

Table 1. Lookup table of rotation angle.

x_i	b_i	$f(x) \geq f(b)$	$\Delta\theta_i$	$s(\alpha_i\ \beta_i)$			
				$\alpha_i\ \beta_i > 0$	$\alpha_i\ \beta_i < 0$	$\alpha_i = 0$	$\beta_i = 0$
0	0	false	0	0	0	0	0
0	0	true	0	0	0	0	0
0	1	false	0	0	0	0	0
0	1	true	0.05π	-1	$+1$	± 1	0
1	0	false	0.01π	-1	$+1$	± 1	0
1	0	true	0.025π	$+1$	-1	0	± 1
1	1	false	0.005π	$+1$	-1	0	± 1
1	1	true	0.025π	$+1$	-1	0	± 1

Where $f(.)$ is the profit; $s(\alpha_i\ \beta_i)$ is the direction sign of rotation angle; and x_i and b_i are the ith bits of the binary solution x and the best solution b, respectively.

Generally speaking, the core concept of the quantum rotation gate is to motivate the probability amplitudes of each qubit in quantum individuals to converge to the corresponding bits of the current best solution in the population. Realistically, the lookup table is a convergence strategy. In this strategy, the fitness *f(x)* of the binary solution *x* after quantum observation is compared with the fitness *f(b)* of the current best solution *b*. The quantum rotation gate will update the probability amplitude (α_i, β_i) toward the direction that favors the emergence of the better solution between *x* and *b*. For example, if $x_i = 0$ and $b_i = 1$, and $f(x_i) > f(b_i)$, then x_i is the current best solution. This means that the state $|0\rangle$ is the optimal state for the *i*th qubit of a quantum individual, and that the QRG will update (α_i, β_i) to increase the probability of the state $|0\rangle$ to make the probability amplitude evolve toward the direction benefiting the appearance of $x_i = 0$. If (α_i, β_i) is located in the first quadrant as shown in Figure 2, the quantum rotation is in a clockwise direction, which favors $x_i = 0$.

Normally, quantum rotation can bring the quantum chromosomes closer to the current optimal chromosomes and generate the exploitation near the current optimal solution to find better solutions. As with other metaheuristic algorithms, individuals of the population converge more closely to the best solution after quantum rotation. The QRG makes the population evolve continuously and speeds up the convergence of the algorithm.

The magnitude of $\Delta\theta_i$ determines the granularity of the search and should be chosen appropriately. A too-small value of the rotation angle will affect the convergence speed and may even lead to a stagnant state. However, if the value is too large, the solutions may diverge or converge prematurely to a local optimal solution [37].

The traditional QRG requires predefined rotation angles, and the value and direction of θ_i should be designed for specific application problems. Because the values of quantum rotation angles are dependent on the problems, the verification of angle selection becomes important, although tedious, work when traditional QRGs are used for optimization problems. If the quantum rotation angles can be obtained adaptively without relying on predefined data, the efficiency of the QRG will be greatly improved, while the types of applications with the QRG can be easily increased. This is exactly what metaheuristic algorithms are good at. Of the large number of metaheuristics, the GWO is a novel swarm optimization algorithm motivated by the social behavior of grey wolves. Because the GWO has the advantages of simple principles, fast searching speed, high seeking accuracy, and easy implementation, it can be easily combined with practical engineering problems [58]. Therefore, the GWO has high theoretical research value and application value, and becomes suitable for generating quantum rotation angles.

In addition, the traditional QRG motivates the probability amplitudes of each qubit in quantum individuals to converge to the corresponding bits of the current best solution in the population. If the current best solution is not the global optimal solution, the direction of quantum rotation may be far from the global optimal solution, and the algorithm may be trapped in local optimal stagnation. Since the GWO records multiple best individuals, it is useful for the QRG to jump out of local optimum with the GWO.

In the proposed QDGWO algorithm, the rotation angle of the QRG is determined with an adaptive GWO. This is described as follows: In the original GWO algorithm, the positions of other ω wolves are updated by the distance vector between themselves and the three best wolves, while for the α, β, and δ wolves, they can hunt the prey more freely. The hunting zone for α, β, and δ wolves will become smaller during the iteration. In the adaptive GWO, these features of the GWO will be inherited and developed.

For a quantum rotation gate with an adaptive GWO (R_{GWO}),$\Delta\theta_{ij}^{t}$ is calculated using the position of α, β, and δ wolves as follows:

$$\Delta\theta_{ij}^{t} = \theta \cdot \left\{ \gamma_i^{\alpha}\left(X_{\alpha j}^{t} - X_{ij}^{t}\right) + \gamma_i^{\beta}\left(X_{\beta j}^{t} - X_{ij}^{t}\right) + \gamma_i^{\delta}\left(X_{\delta j}^{t} - X_{ij}^{t}\right) \right\} \tag{18}$$

where $i \in \{1,2,3,\ldots,n\}$; $j \in \{1,2,3,\ldots,m\}$; n is the size of the population; m is the number of items; θ is the rotation angle magnitude; X_{ij}^{t} is the jth component in the binary individual of the ith wolf during tth iteration; and γ_i^{α}, γ_i^{β} and γ_i^{δ} are determined by comparing the fitness of the current binary individual with α, β, and δ wolves as follows:

$$\gamma_i^{\alpha} = \begin{cases} \frac{f(X_{\alpha}^{t})}{f(X_i^{t})}, \text{ if } f(X_i^{t}) < f(X_{\alpha}^{t}) \\ \frac{N(0,1)\times t_{\max}}{k(t_{\max}+t)}, \text{ otherwise} \end{cases} \tag{19}$$

$$\gamma_i^{\beta} = \begin{cases} \frac{f\left(X_{\beta}^{t}\right)}{f(X_i^{t})}, \text{ if } f(X_i^{t}) < f\left(X_{\beta}^{t}\right) \\ \frac{N(0,1)\times t_{\max}}{k(t_{\max}+t)}, \text{ otherwise} \end{cases} \tag{20}$$

$$\gamma_i^{\delta} = \begin{cases} \frac{f(X_{\delta}^{t})}{f(X_i^{t})}, \text{ if } f(X_i^{t}) < f(X_{\delta}^{t}) \\ \frac{N(0,1)\times t_{\max}}{k(t_{\max}+t)}, \text{ otherwise} \end{cases} \tag{21}$$

where N(0,1) is the Gaussian distribution, $\mu = 0, \sigma = 1$.

For the ω wolves, they hunt the prey based on their own positions vs. the positions of the three best wolves. It is obvious that for the ith wolf in the ω wolves group, $f(X_i^{t}) < f(X_{\delta}^{t}) \leq f\left(X_{\beta}^{t}\right) \leq f(X_{\alpha}^{t})$. Therefore, the rotation angle of the ith wolf can be calculated with the binary individuals of α, β, δ, and the ith wolf.

For α, β, and δ wolves, they have the duty of searching for the optimal solution from their old positions. With increasing t, γ_i^{α}, γ_i^{β} and γ_i^{δ} will be much smaller in the late stages of the iteration than what they are in the early stages of the iteration, which helps the ω wolves converge toward an estimated position of prey calculated by α, β, and δ wolves. This strategy is helpful for jumping out of local optimal stagnation, especially in the final stage of iterations.

The speed of convergence and quality of the solution are greatly affected by the magnitude of the rotation angle θ, which is given by:

$$\begin{array}{l} \theta = \theta_{\min} + \left(1 - \frac{t}{t_{\max}}\right) \cdot (\theta_{\max} - \theta_{\min}) \\ 0 < \theta_{\min} < \theta_{\max} \end{array} \tag{22}$$

where θ is linearly decreasing from $\theta_{\max}$ to $\theta_{\min}$ during the iteration.

In the end, $q_{\theta i}^{t+1}$ can be generated by:

$$\theta_{ij}^{t+1} = \theta_{ij}^{t} + s_{ij}^{t} \Delta\theta_{ij}^{t} \tag{23}$$

where s_{ij}^{t} is the direction signal of the rotation angle as follows:

$$s_{ij}^{t} = \begin{cases} 1, \text{ if } \theta_{ij}^{t} \in (0, \frac{\pi}{2}) \cup (\pi, \frac{3\pi}{2}) \\ -1, \text{ if } \theta_{ij}^{t} \in (\frac{\pi}{2}, \pi) \cup (\frac{3\pi}{2}, 2\pi) \\ \pm 1, \text{ otherwise} \end{cases} \tag{24}$$

With the adaptive strategy of γ and θ, the searching granularity of the QDGWO changes from coarse to fine. A different searching granularity in the iteration will facilitate the process for search agents to reach the global optimal solution.

3.9. Procedure of QDGWO Algorithm

Based on the description above, the procedure of the QDGWO and its main steps can be summarized as shown in Algorithm 2.

Algorithm 2 QDGWO

```
$t \leftarrow 0$// Initializes the iteration
Initialize $Q(0)$ by Equations (5) and (6)
while ($t$ < MaxIter) do
  Observe to get $X(t)$ from $q(t)$//Quantum observation
  Evaluate fitness of $X(t)$ by Equation (7)
  Apply mutation on $q^M(t)$ by Equations (8) and (11)//Adaptive mutation
  Obtain $q^C(t)$ by crossover by Equations (12) and (13)//Crossover
  Observe to get $X^C(t)$ from $q^C(t)$//Quantum observation
  Evaluate fitness of $X^C(t)$
  if the trial binary individuals $X^C(t)$ is better than $X(t)$ then
    Update $X(t+1)$ and $q(t+1)$ by Equations (14) and (15)//Selection
  else
    Update $q(t + 1)$ using QRG with adaptive GWO by Equations (16)–(24)
  end if
  $t \leftarrow t + 1$
end while
```

3.10. Example of QDGWO Algorithm to Solve KP01

In the following, a KP01 problem is described as an example to be solved by the QDGWO algorithm.

Given a set of ten items, $W = (x_1, x_2, x_3, \ldots, x_{10})$, and w_i is the weight of the ith item x_i; p_i is the value of x_i; and C is the weight capacity of the knapsack. Suppose that $w_i = i$, and $p_i = w_i+5$, and $C = \frac{1}{2} \sum_{i=1}^{m} w_i = 27.5$.

Then, this 0-1 knapsack problem can be defined as:

$$\begin{aligned} &\text{Maximize } f(x) = \sum_{i=1}^{m} p_i x_i \\ &\text{s.t. } \sum_{i=1}^{m} w_i x_i \leq 27.5, x_i \in \{0,1\}, \forall i \in \{1,2,\ldots,10\} \end{aligned} \tag{25}$$

where $w_i = i$, and $p_i = w_i + 5 = i + 5$.

In this example, the population size was set to 20, and the maximum number of iterations was set to 200. The other parameters are presented in Table 2. The evolution process of q_1, an individual of the quantum population, is shown below.

Table 2. Parameters of algorithms in experiments.

	QEA	AQDE	QSE	QDGWO
Differential control parameter (F)	/	rand(0,1) × rand(0,1) × 0.1	/	F_0 = 0.02 F_1 = 0.03
Crossover control parameter (CR)	/	Gaussian distribution N(0.5, 0.0375)	/	Gaussian distribution N(0.5, 0.0375)
Parameters of PSO	/	/	W = 0.7298 c_1 = 1.42 c_2 = 1.57	/
Quantum rotation angle ($\Delta\theta$)	0.01π	/	/	$\theta_{\min} = 0.01\pi$ $\theta_{\max} = 0.03\pi$ $k = 10$

The initial quantum population $Q(0) = \left(q_1^0, q_2^0, \ldots, q_{20}^0\right)$ was initialized by Equations (5) and (6). The vector $q_{\theta 1}$ was initialized by $q_{\theta 1}^0 = \left(\frac{3\pi}{4}, \frac{\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}, \frac{3\pi}{4}, \frac{\pi}{4}, \frac{5\pi}{4}, \frac{\pi}{4}\right)$, and the quantum individual q_1^0 was initialized by $q_1^0 = \left[\begin{array}{c|c|c|c|c|c|c|c|c|c} \frac{-\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{array}\right]$.

After quantum observation, the binary individual X_1^0 was generated as $(1,0,1,1,0,0,1,0,0,1)$, and the fitness of X_1^0 was evaluated as $f\left(X_1^0\right) = \sum_{i=1}^{10} p_i x_i^0 = 50$.

The mutation vector $q_{\theta 1}^{\mathrm{M}0}$ was generated by Equations (8)–(11), where t = 0; $\omega = e^{1-\frac{t_{\max}}{t_{\max}-t}} = 1$; and $F^0 = F_0 + F_1 \cdot 2^{\omega} \cdot \mathrm{rand}(0,1) = 0.02 + 0.03 \times 2 \times 0.68 = 0.0608$ (rand(0,1) was randomly generated as 0.68).

The vector $q_{\theta\alpha}$ corresponding to the current best binary individual X_α and two different random target vectors $q_{\theta r1}$ and $q_{\theta r2}$ are shown as:

$$q_{\theta\alpha} = \left(\frac{\pi}{4}, \frac{3\pi}{4}, \frac{\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}, \frac{3\pi}{4}, \frac{\pi}{4}, \frac{\pi}{4}, \frac{3\pi}{4}, \frac{3\pi}{4}\right);$$

$$q_{\theta r1} = \left(\frac{3\pi}{4}, \frac{7\pi}{4}, \frac{7\pi}{4}, \frac{5\pi}{4}, \frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{5\pi}{4}, \frac{\pi}{4}, \frac{7\pi}{4}\right);$$

$$q_{\theta r2} = \left(\frac{7\pi}{4}, \frac{7\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{\pi}{4}, \frac{5\pi}{4}, \frac{\pi}{4}, \frac{\pi}{4}, \frac{5\pi}{4}, \frac{3\pi}{4}\right).$$

The quantum mutation vector $q_{\theta 1}^{\mathrm{M}0}$ was generated by:

$$\begin{aligned} q_{\theta 1}^{\mathrm{M}0} &= q_{\theta\alpha} + F^0(q_{\theta r1} - q_{\theta r2}) \\ &= (0.1892\pi, 0.75\pi, 0.3108\pi, 1.25\pi, 1.75\pi, 0.7196\pi, 0.3108\pi, 0.3108\pi, 0.6892\pi, 0.8108\pi) \end{aligned}$$

The quantum trial vector $q_{\theta 1}^{C0}$ at the tth generation was generated by Equations (12) and (13):

$$q_{\theta 1}^{C0} = (0.75\pi, 0.75\pi, 0.3108\pi, 1.75\pi, 1.25\pi, 1.75\pi, 0.3108\pi, 0.25\pi, 0.6892\pi, 0.25\pi)$$

where *CR* was randomly generated as 0.35, and *rnbr_i* = 3.

After quantum observation, the trial binary individual X_1^{C0} was generated as $(1,0,1,1,0,0,1,0,1,0)$, and the fitness was evaluated as $f\left(X_1^{C0}\right) = \sum_{i=1}^{10} p_i x_i^0 = 49$. Because $f\left(X_1^{Ct}\right) < f\left(X_1^t\right)$, q^{t+1} should be updated by the QRG with an adaptive GWO.

The positions and profits of the current three best wolves were shown as:

$$X_\alpha^0 = (1,1,1,0,1,1,0,1,0,0), f(X_\alpha^0) = \sum_{i=1}^{10} p_i x_i^0 = 55;$$

$$X_\beta^0 = (1,1,1,1,1,0,0,0,1,0), f(X_\beta^0) = \sum_{i=1}^{10} p_i x_i^0 = 54;$$

$$X_\delta^0 = (0,1,1,0,0,1,1,0,1,0), f(X_\delta^0) = \sum_{i=1}^{10} p_i x_i^0 = 52.$$

Additionally, $\theta = \theta_{\min} + \left(1 - \frac{t}{t_{\max}}\right) \cdot (\theta_{\max} - \theta_{\min}) = \theta_{\max} = 0.03\pi$; $\gamma_1^\alpha = \frac{f\left(X_\alpha^0\right)}{f\left(X_1^0\right)} = 1.1$; $\gamma_1^\beta = \frac{f(X_\beta^0)}{f(X_1^0)} = 1.08$; $\gamma_1^\delta = \frac{f(X_\delta^0)}{f(X_1^0)} = 1.04$.

Then, the quantum individual of the next iteration $q_{\theta i}^{t+1}$ was generated by Equations (16)–(24) as:

$$\begin{aligned}\Delta q_{\theta 1}^0 &= [\Delta\theta_{11}^0, \Delta\theta_{12}^0, \ldots, \Delta\theta_{110}^0] \\ &= [-0.0312\pi, 0.0966\pi, 0, -0.0642\pi, 0.0654\pi, 0.0642\pi, -0.0654\pi, 0.033\pi, 0.0636\pi, -0.0966\pi]\end{aligned}$$

$$q_{\theta 1}^0 = \left(\frac{3\pi}{4}, \frac{\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}, \frac{3\pi}{4}, \frac{\pi}{4}, \frac{5\pi}{4}, \frac{\pi}{4}\right)$$

$$\begin{aligned}q_{\theta 1}^1 &= q_{\theta 1}^0 + s_1^0 \Delta q_{\theta 1}^0 \\ &= (0.7812\pi, 0.3466\pi, 1.25\pi, 1.8142\pi, 1.3154\pi, 1.6858\pi, 0.8154\pi, 0.283\pi, 1.3136\pi, 0.1534\pi)\end{aligned}$$

The individuals of the next iteration continued to evolve until the maximum number of iterations was reached. After the iterations, the best profit of this KP01 problem was 57, and one of the best solutions was (0,1,1,1,1,1,1,0,0,0).

4. Experimental Results

To assess the performance of the proposed QDGWO algorithm, two groups of datasets are used for solving the KP01.

All experiments were conducted with Matlab 2016b, running on an Intel Core i7-4790 CPU @ 3.60 GHz, and Windows 7 Ultimate Edition.

In the first experiment described in [37], there were 50, 250, 500, 1000, 1500, 2000, 2500, and 3000 dimension sets of data by Equation (26) to test the performance of the QDGWO in high-dimension situations.

Given a set of m items, $W = (x_1, x_2, x_3, \ldots, x_m)$.

$$\begin{aligned} w_i &= \text{rand\_i}[1,\ 10] \\ p_i &= w_i + 5 \\ C &= \tfrac{1}{2} \sum_{i=1}^{m} w_i \end{aligned} \tag{26}$$

where w_i is the weight of the ith item x_i; p_i is the value of x_i; C is the weight capacity of the knapsack; and m is the number of items.

In the first experiment, m ranged from 50 to 3000, and the maximum number of iterations in all cases was set to 1000.

To verify the effectiveness and efficiency of the QDGWO, the results of the proposed algorithm were compared with three algorithms: QEA [37], AQDE [45], and QSE [48]. The parameters of algorithms used in the experiments are presented in Table 2, where the population size is 20. The best profits, the average profits, the worst profits, and the standard deviations of 30 independent runs are shown in Table 3 and Figures 4–7. The Wilcoxon signed-rank test [59] is performed for the results of the competing algorithms in Table 3 with a significance level $\alpha = 0.05$, where +, −, and = indicate that this algorithm is superior, inferior, or equal to the QDGWO, respectively.

Table 3. Experimental results for 0-1 knapsack problems (Experiment 1).

Number of Items		QEA	AQDE	QSE	QDGWO
50	Best	302(=)	292(−)	297(=)	302
	Average	300.63(=)	287.2(−)	294.26(−)	302
	Worst	297(=)	282(−)	290(−)	302
	Std	2.23(−)	2.97(−)	2.41(−)	0
250	Best	1517(=)	1417(−)	1446(−)	1554
	Average	1502.9(=)	1397.6(−)	1,427.7(−)	1549.3
	Worst	1496(=)	1382(−)	1412(−)	1542
	Std	4.4562(−)	7.3178(−)	8.2040(−)	2.3419
500	Best	2946(=)	2772(−)	2799(−)	3091
	Average	2917.3(−)	2732(−)	2783(−)	3072.1
	Worst	2907(−)	2717(−)	2763(−)	3058
	Std	8.8198(=)	11.3304(=)	9.2364(=)	8.9624
1000	Best	5695(−)	5382(−)	5460(−)	6121
	Average	5662.5(−)	5,364.4(−)	5442.2(−)	6085.3
	Worst	5633(−)	5342(−)	5422(−)	6048
	Std	12.7028(=)	11.2975(=)	10.1018(+)	13.4812
1500	Best	8464(−)	8198(−)	8128(−)	9126
	Average	8,439.4(−)	8,178.7(−)	8,082.8(−)	9077.1
	Worst	8414(−)	8149(−)	8039(−)	9027
	Std	15.1535(+)	13.6188(+)	20.6722(=)	21.5347
2000	Best	11,217(−)	10,951(−)	10,813(−)	12,027
	Average	11,191.2(−)	10,900.4(−)	10,781.1(−)	11,971.4
	Worst	11,164(=)	10,865(−)	10,747(−)	11,913
	Std	14.7202(+)	24.6167(=)	16.2827(+)	24.3967
2500	Best	13,907(−)	13,569(−)	13,466(−)	14,886
	Average	13,865.8(−)	13,523.3(−)	13,394.2(−)	14,831.4
	Worst	13,839(−)	13,482(−)	13,342(−)	14,751
	Std	19.0504(+)	24.5971(+)	23.5438(+)	28.4894
3000	Best	16,604(−)	16,221(−)	16,071(−)	17,769
	Average	16,549.9(−)	16,175.8(−)	16,033.2(−)	17,670.1
	Worst	16,506(−)	16,128(−)	15,995(−)	17,588
	Std	20.2286(+)	22.0621(+)	20.5269(+)	29.5280

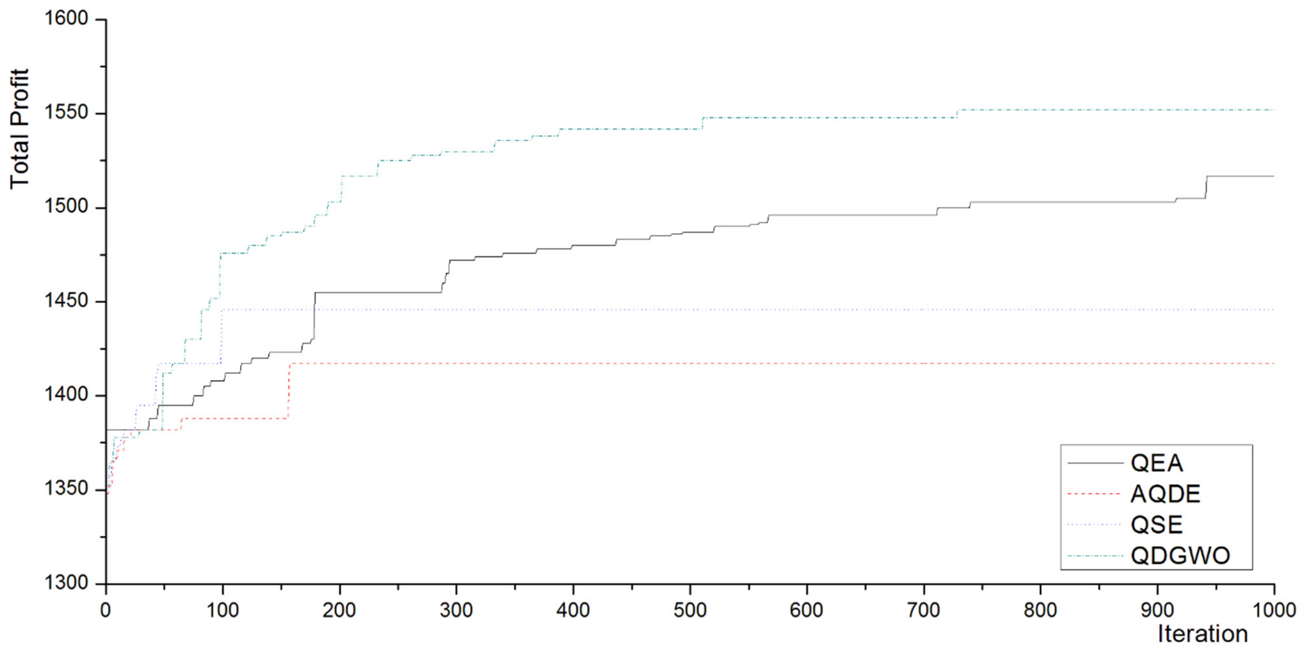

Figure 4. Best profits for the 0-1 knapsack problems (250 items in Experiment 1).

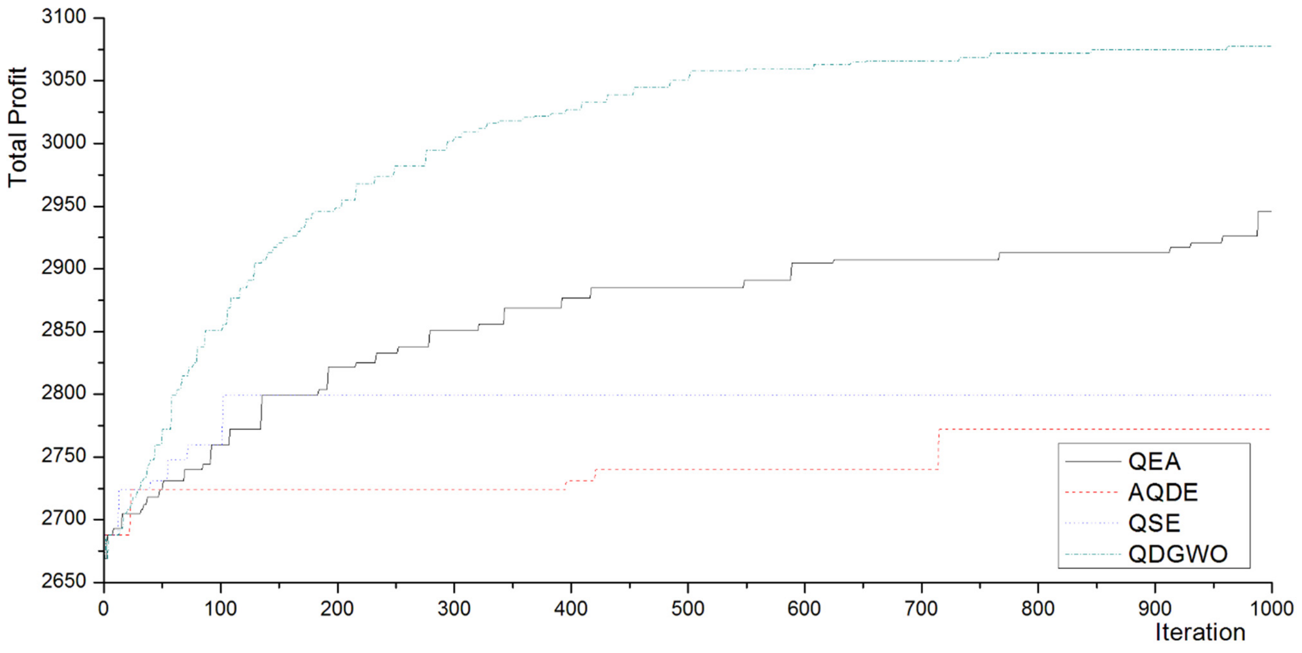

Figure 5. Best profits for the 0-1 knapsack problems (500 items in Experiment 1).

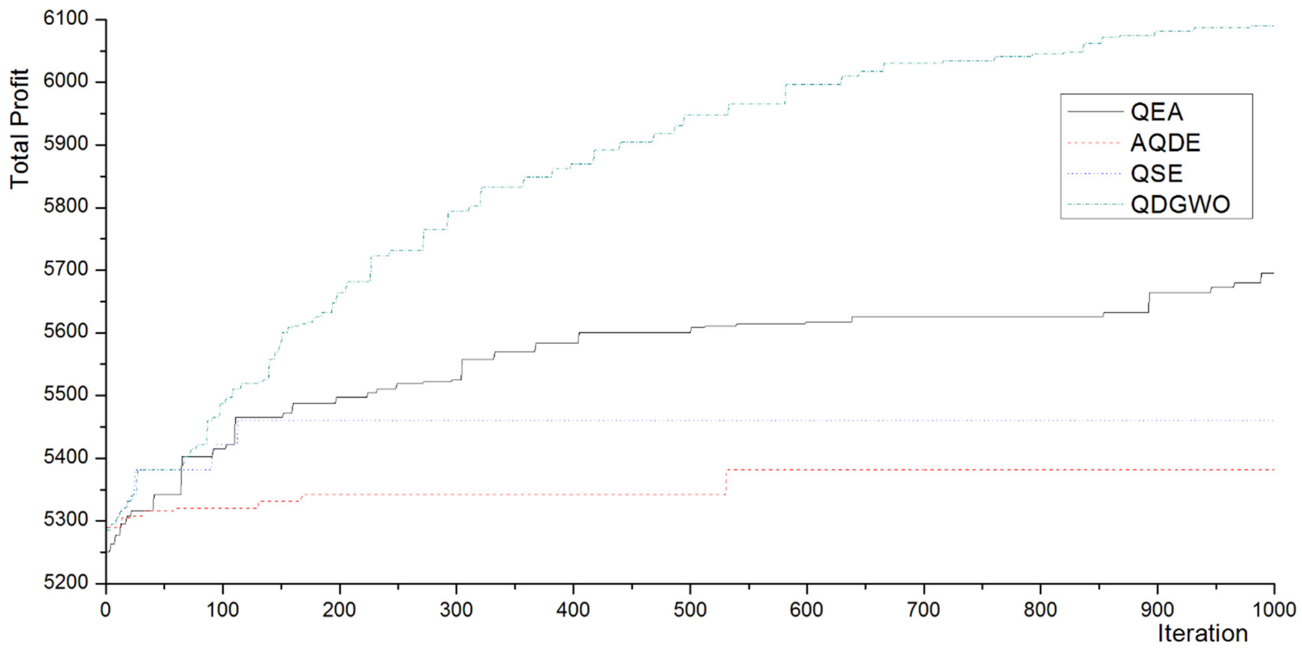

Figure 6. Best profits for the 0-1 knapsack problems (1000 items in Experiment 1).

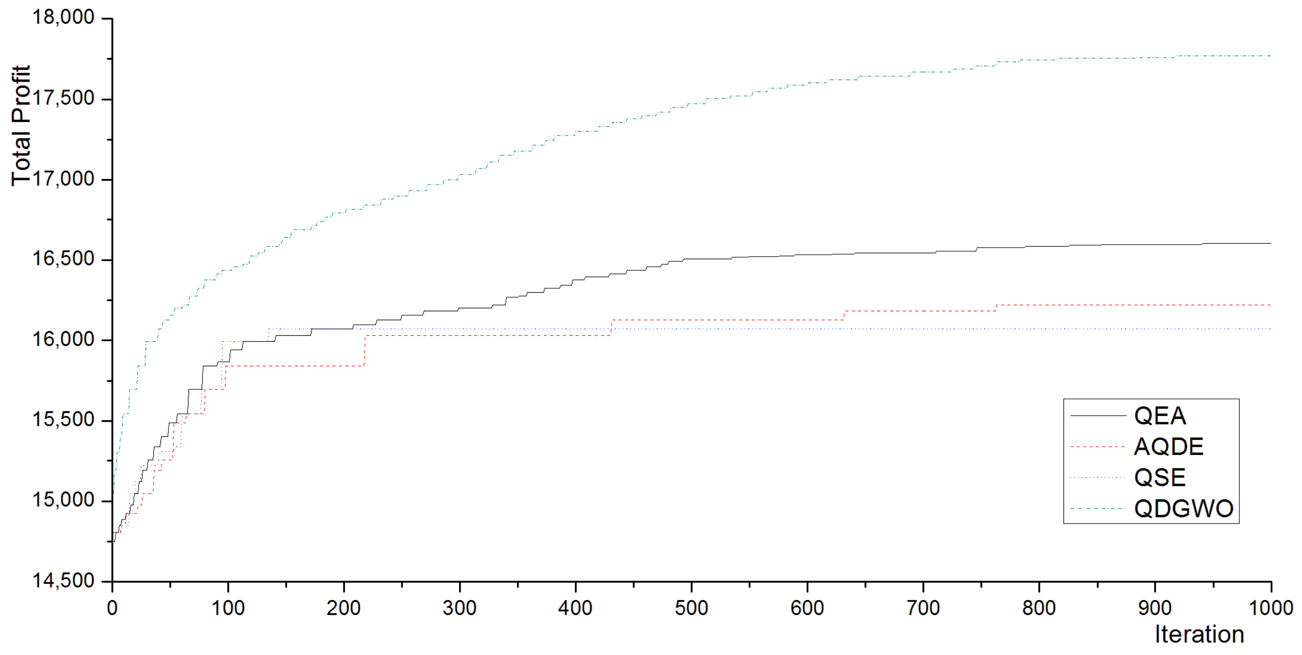

Figure 7. Best profits for the 0-1 knapsack problems (3000 items in Experiment 1).

To illustrate the importance of the role of the crossover operation of the DE in exploring the global optimum, comparative tests between the QDGWO with and without the crossover operation were performed. Moreover, we compared the binomial crossover operator of the DE with the exponential crossover operator of the DE. The best profits, the average profits, the worst profits, and the standard deviations of 30 independent runs are presented in Table 4. The Wilcoxon signed-rank test [59] is performed for the results in Table 4 with a significance level $\alpha = 0.05$, where +, −, and = indicate that this strategy is superior, inferior, or equal to the QDGWO with a binomial crossover of the DE, respectively.

Table 4. Experimental results of QDGWO algorithm without crossover of DE, with binomial crossover of DE, and with exponential crossover of DE.

Number of Items		**Without Crossover of DE**	**With Binomial Crossover of DE (*CR* = 0.5)**	**With Exponential Crossover of DE (*CR* = 0.5)**
500	Best	3001(−)	3046	3046(=)
	Average	2990.3(−)	3038.6	3040.1(=)
	Worst	2981(−)	3031	3031(=)
	Std	6.1752(−)	3.6191	3.4287(=)
1000	Best	5926(−)	6126	6126(=)
	Average	5893.8(−)	6109.4	6107.7(=)
	Worst	5851(−)	6096	6081(=)
	Std	18.0843(−)	7.2612	9.5447(=)
1500	Best	8752(−)	9126	9126(=)
	Average	8707.7(−)	9094.9	9090.9(=)
	Worst	8647(−)	9066	9042(=)
	Std	23.2206(−)	16.8516	21.4710(−)

In the second experiment described in [49], there were 50, 200, 500, 1000, 1500, and 2000 dimension sets of data by Equation (27) to test the performance of the QDGWO in high-dimension situations.

Given a set of m items, $W = (x_1, x_2, x_3, \ldots, x_m)$.

$$\begin{aligned} &w_i = \text{rand\_i}[1,\ 10] \\ &p_i = w_i + 5 \\ &C = \tfrac{3}{4}\sum_{i=1}^{m} w_i \end{aligned} \tag{27}$$

where w_i is the weight of the ith item x_i; p_i is the value of x_i; C is the weight capacity of the knapsack; and m is the number of items.

In the second experiment, m ranged from 50 to 2000, and the maximum number of iterations in all cases was set to 1000.

To present the performance of the proposed algorithm in the global optimization, we compared the QDGWO algorithm with the QIHSA [49] for knapsack problems. The optimization results of the success rate ($SR\%$) and the best profit are shown in Table 5. The Wilcoxon signed-rank test [59] is performed for the results of the QIHSA in Table 5 with a significance level α = 0.05, where +, −, and = indicate that this algorithm is superior, inferior, or equal to the QDGWO, respectively.

The obtained results demonstrate the competitive performance of the proposed QDGWO algorithm. According to the results, the proposed algorithm is more efficient for the high-dimensional 0-1 knapsack problems, as shown in Table 4 and Figures 3–5. Compared with the QEA [25], AQDE [33], QSE [36], and QIHSA [37], the QDGWO was the most effective and efficient algorithm in the experiments. The advantages of the QDGWO became more obvious when the number of items was large, especially in high dimensional cases of the knapsack problems.

Table 5. Experimental results of QDGWO and QIHSA for 0-1 knapsack problems (Experiment 2).

Test	Item Size	Optimal Solution		QIHSA	QDGWO

Knapinst50	50	1177	SR%
			best
				99.83(=)
				1175(=)
					100
					1177

Knapinst200	200	4860	SR%
			best
				97.83(−)
				4755(−)
					100
					4860

Knapinst500	500	11,922	SR% best	93.74(−) 11,174(−)	98.56 11,748
Knapinst1000	1000	24,356	SR% best	87.97(−) 21,427(−)	98.14 23,903
Knapinst1500	1500	35,891	SR% best	86.31(−) 30,978(−)	97.25 34,904
Knapinst2000	2000	49,007	SR% best	85.8(−) 42,052(−)	96.36 47,223

The proposed algorithm obtains both rapid exploration and high exploitation in searching solutions. The QDGWO converges quickly to the global optimal solution. For example, the algorithm approaches the global optimum at about the 500th iteration in the case of 500 items (see Figure 4). However, the algorithm continues searching near the global optimal solution, i.e., the exploitation. To illustrate this, in the case of 500 items (see Figure 4), the QDGWO continues seeking further optimization after approaching the optimal solution and obtains better solutions in the exploitation until the end of the iterations.

Based on the results shown in Table 3, it can be concluded that the crossover operation plays a significant role in searching the solution space efficiently. However, the performance of the QDGWO is not very sensitive to which kind of crossover operator is used in the algorithm. From the experiment results, the binomial crossover operator of the DE yields slightly better optimal solutions than the exponential crossover operator of the DE in all cases. The results can be interpreted to show that quantum updating with the quantum rotation gate remains the most decisive and crucial operation in exploring the search space even if the crossover operation is required to improve the solutions.

Finally, compared with the other four methods, the experimental results show the advantages of the collaborative optimization with operations of adaptive mutation, crossover, and quantum rotation gate with the adaptive GWO in investigating the search space.

5. Conclusions

A quantum-inspired differential evolution algorithm with grey wolf optimizer (QDGWO) was proposed to solve the 0-1 knapsack problems. The proposed algorithm combined the superposition principles of quantum computing, differential evolution operations, and the hunting behaviors of grey wolves. The QDGWO used the principles of quantum computing such as quantum superposition states and quantum gates. Furthermore, it contained mutation, crossover, and selection operations of the DE. To maintain a better balance between the exploration and exploitation of searching for the global optimal solution, the proposed algorithm adapted a quantum rotation gate with the adaptive GWO to update the population of solutions. The results of tests performed for resolving the knapsack problems demonstrate that the QDGWO was able to enhance diversity and convergence performance for solving 0-1 knapsack problems. In addition, the QDGWO was effective and efficient in finding the optimal solutions for high-dimensional situations.

Although the QDGWO displays excellent performance in solving 0-1 knapsack problems, there are several directions of improvement for the proposed algorithm. First, to improve the effectiveness of the QDGWO, initial solutions of the quantum population can be generated with metaheuristic methods. In addition, the proposed approaches can be applied to solve other combinatorial optimization problems. Moreover, it is worth studying how to use the concepts of quantum computing in other novel metaheuristic approaches such as the MPA [22] and AOA [27], as well as multi-objective optimization algorithms.

Author Contributions: Conceptualization, Y.W. and W.W.; methodology, Y.W.; software, Y.W.; validation, Y.W.; formal analysis, Y.W.; investigation, Y.W.; resources, Y.W.; data curation, Y.W.; writing—original draft preparation, Y.W.; writing—review and editing, Y.W. and W.W.; visualization, Y.W.; supervision, W.W.; funding acquisition, W.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No. 61873240).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the anonymous reviewers for their constructive comments and suggestions. This work was supported in part by the National Natural Science Foundation of China (No. 61873240).

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

References

1. Kellerer, H.; Pferschy, U.; Pisinger, D. *Knapsack Problems*; Springer: Berlin/Heidelburg, Germany, 2004.
2. Wang, X.; He, Y. Evolutionary algorithms for knapsack problems. *J. Softw.* **2017**, *28*, 1–16.
3. Jourdan, L.; Basseur, M.; Talbi, E.G. Hybridizing exact methods and metaheuristics: A taxonomy. *Eur. J. Oper. Res.* **2009**, *199*, 620–629. [CrossRef]
4. Shih, W. A branch and bound method for the multiconstraint zero-one knapsack problem. *J. Oper. Res. Soc.* **1979**, *30*, 369–378. [CrossRef]
5. Toth, P. Dynamic programming algorithms for the zero-one knapsack problem. *Computing* **1980**, *25*, 29–45. [CrossRef]
6. Zou, D.; Gao, L.; Li, S.; Wu, J. Solving 0-1 knapsack problem by a novel global harmony search algorithm. *Appl. Soft Comput.* **2011**, *11*, 1556–1564. [CrossRef]
7. Fogel, D.B. Introduction to evolutionary computation. In *Evolutionary Computation 1*; Taylor & Francis Group: New York, NY, USA, 2000.
8. Chen, G.-L.; Wang, X.-F.; Zhuang, Z.-Q.; Wang, D.-S. *Genetic Algorithm and Its Applications*; The People's Posts and Telecommunications Press: Beijing, China, 2003.
9. Dorigo, M.; Birattari, M.; Stutzle, T. Ant colony optimization. *IEEE Comput. Intell. Mag.* **2006**, *1*, 28–39. [CrossRef]
10. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [CrossRef]
11. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]
12. Yang, X.-S.; Deb, S. Cuckoo search via Lévy flights. In Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, India, 9–11 December 2009; pp. 210–214.
13. Yang, X.-S.; Xin, S. Firefly algorithm, stochastic test functions and design optimisation. *Int. J. Bio-Inspired Comput.* **2010**, *2*, 78–84. [CrossRef]
14. Feng, Y.-H.; Wang, G.-G.; Wang, L. Solving randomized time-varying knapsack problems by a novel global firefly algorithm. *Eng. Comput.* **2018**, *34*, 621–635. [CrossRef]
15. Cao, J.; Yin, B.; Lu, X.; Kang, Y.; Chen, X. A modified artificial bee colony approach for the 0-1 knapsack problem. *Appl. Intell.* **2017**, *48*, 1582–1595. [CrossRef]
16. Wu, H.; Zhou, Y.; Luo, Q. Hybrid symbiotic organisms search algorithm for solving 0-1 knapsack problem. *Int. J. Bio-Inspired Comput.* **2018**, *12*, 23–53. [CrossRef]
17. Feng, Y.-H.; Jia, K.; He, Y.-C. An improved hybrid encoding cuckoo search algorithm for 0-1 knapsack problems. *Comput. Intell. Neurosci.* **2014**, *2014*, 970456. [CrossRef]
18. Zhou, Y.-Q.; Chen, X.; Zhou, G. An improved monkey algorithm for a 0-1 knapsack problem. *Appl. Soft Comput.* **2016**, *38*, 817–830. [CrossRef]
19. Sun, J.; Miao, Z.; Gong, D.-W.; Zeng, X.-J.; Li, J.-Q.; Wang, G.-G. Interval multi-objective optimization with memetic algorithms. *IEEE Trans. Cybern.* **2020**, *50*, 3444–3457. [CrossRef]
20. Wang, G.-G.; Guo, L.-H.; Gandomi, A.H.; Hao, G.-S.; Wang, H.-Q. Chaotic krill herd algorithm. *Inf. Sci.* **2014**, *274*, 17–34. [CrossRef]
21. Wang, G.-G.; Bai, D.; Gong, W.; Ren, T.; Liu, X.; Yan, X. Particle-swarm krill herd algorithm. In Proceedings of the 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Bangkok, Thailand, 16–19 December 2018; pp. 1073–1080.

22. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine predators algorithm: A nature-inspired metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [CrossRef]
23. Wang, G.-G. Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Comput.* **2016**, *10*, 151–164. [CrossRef]
24. Feng, Y.-H.; Wang, G.-G. Binary moth search algorithm for discounted {0-1} knapsack problem. *IEEE Access* **2018**, *6*, 10708–10719. [CrossRef]
25. Feng, Y.-H.; Yi, J.-H.; Wang, G.-G. Enhanced moth search algorithm for the set-union knapsack problems. *IEEE Access* **2019**, *7*, 173774–173785. [CrossRef]
26. Gao, D.; Wang, G.-G.; Pedrycz, W. Solving fuzzy job-shop scheduling problem using DE algorithm improved by a selection mechanism. *IEEE Trans. Fuzzy Syst.* **2020**, *28*, 3265–3275. [CrossRef]
27. Abualigah, L.; Diabat, A.; Mirjalili, S.; Abd Elaziz, M.; Gandomi, A.H. The arithmetic optimization algorithm. *Comput. Methods Appl. Mech. Eng.* **2021**, *376*, 113609. [CrossRef]
28. Wang, G.-G.; Deb, S.; Cui, Z. Monarch butterfly optimization. *Neural Comput. Appl.* **2019**, *31*, 1995–2014. [CrossRef]
29. Feng, Y.-H.; Wang, G.-G.; Deb, S.; Lu, M.; Zhao, X.-J. Solving 0-1 knapsack problem by a novel binary monarch butterfly optimization. *Neural Comput. Appl.* **2017**, *28*, 1619–1634. [CrossRef]
30. Feng, Y.-H.; Wang, G.-G.; Li, W.-B.; Li, N. Multi-strategy monarch butterfly optimization algorithm for discounted {0-1} knapsack problem. *Neural Comput. Appl.* **2018**, *30*, 3019–3036. [CrossRef]
31. Feng, Y.-H.; Wang, G.-G.; Dong, J.-Y.; Wang, L. Opposition-based learning monarch butterfly optimization with Gaussian perturbation for large-scale 0-1 knapsack problem. *Comput. Electr. Eng.* **2018**, *67*, 454–468. [CrossRef]
32. Feng, Y.-H.; Yu, X.; Wang, G.-G. A novel monarch butterfly optimization with global position updating operator for large-scale 0-1 knapsack problems. *Mathematics* **2019**, *7*, 1056. [CrossRef]
33. Wang, G.-G.; Tan, Y. Improving metaheuristic algorithms with information feedback models. *IEEE Trans. Cybern.* **2019**, *49*, 542–555. [CrossRef]
34. Benioff, P. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *J. Stat. Phys.* **1980**, *22*, 563–591. [CrossRef]
35. Feynman, R.P. Simulating physics with computers. *Int. J. Theor. Phys.* **1999**, *21*, 467–488. [CrossRef]
36. Han, K.-H.; Kim, J.-H. Genetic quantum algorithm and its application to combinatorial optimization problems. In Proceedings of the International Congress on Evolutionary Computation (CEC2000), San Diego, CA, USA, 16–19 July 2000; Volume 2, pp. 1354–1360.
37. Han, K.-H.; Kim, J.-H. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. *IEEE Trans. Evol. Comput.* **2002**, *6*, 580–593. [CrossRef]
38. Talbi, H.; Draa, A.; Batouche, M. A new quantum-inspired genetic algorithm for solving the travelling salesman problem. In Proceedings of the 2004 IEEE International Conference on Industrial Technology, 2004, IEEE ICIT'04, Hammamet, Tunisia, 8–10 December 2004; Volume 3, pp. 1192–1197.
39. Chang, C.-C.; Chen, C.-Y.; Fan, C.-W.; Chao, H.-C.; Chou, Y.-H. Quantum-inspired electromagnetism-like mechanism for solving 0/1 knapsack problem. In Proceedings of the 2010 2nd International Conference on Information Technology Convergence and Services, Cebu, Philippines, 11–13 August 2010; pp. 1–6.
40. Xiong, H.; Wu, Z.; Fan, H.; Li, G.; Jiang, G. Quantum rotation gate in quantum-inspired evolutionary algorithm: A review, analysis and comparison study. *Swarm Evol. Comput.* **2018**, *42*, 43–57. [CrossRef]
41. Zhou, W.; Zhou, C.; Liu, G.; Lv, H.; Liang, Y. An improved quantum-inspired evolutionary algorithm for clustering gene expression data. In *Computational Methods*; Springer: Dordrecht, The Netherlands, 2006; pp. 1351–1356.
42. Xiao, J.; Yan, Y.; Lin, Y.; Yuan, L.; Zhang, J. A quantum-inspired genetic algorithm for data clustering. In Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), Hong Kong, China, 1–6 June 2008; pp. 1513–1519.
43. Han, K.-H.; Kim, J.-H. Quantum-inspired evolutionary algorithms with a new termination criterion, H^{ε}Gate, and two-phase scheme. *IEEE Trans. Evol. Comput.* **2004**, *8*, 156–169. [CrossRef]
44. Storn, R.; Price, K. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
45. Hota, A.R.; Pat, A. An adaptive quantum-inspired differential evolution algorithm for 0-1 knapsack problem. In Proceedings of the 2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC), Kitakyushu, Japan, 15–17 December 2010; pp. 703–708.
46. Draa, A.; Meshoul, S.; Talbi, H.; Batouche, M. A quantum-inspired differential evolution algorithm for solving the N-queens problem. *Neural Netw.* **2011**, *1*, 21–27.
47. Su, H.; Yang, Y. Quantum-inspired differential evolution for binary optimization. In Proceedings of the 2008 Fourth International Conference on Natural Computation, Jinan, China, 18–20 October 2008; Volume 1, pp. 341–346.
48. Wang, Y.; Feng, X.-Y.; Huang, Y.-X.; Pu, D.-B.; Zhou, W.-G.; Liang, Y.-C.; Zhou, C.-G. A novel quantum swarm evolutionary algorithm and its applications. *Neurocomputing* **2007**, *70*, 633–640. [CrossRef]
49. Layeb, A. A hybrid quantum inspired harmony search algorithm for 0-1 optimization problems. *J. Comput. Appl. Math.* **2013**, *253*, 14–25. [CrossRef]

50. Zouache, D.; Moussaoui, A. Quantum-inspired differential evolution with particle swarm optimization for knapsack problem. *J. Inf. Sci. Eng.* **2015**, *31*, 1757–1773.
51. Gao, Y.; Zhang, F.; Zhao, Y.; Li, C. Quantum-inspired wolf pack algorithm to solve the 0-1 knapsack problem. *Math. Probl. Eng.* **2018**, *2018*, 5327056. [CrossRef]
52. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
53. Srikanth, K.; Panwar, L.K.; Panigrahi, B.K.; Herrera-Viedma, E.; Sangaiah, A.K.; Wang, G.-G. Meta-heuristic framework: Quantum inspired binary grey wolf optimizer for unit commitment problem. *Comput. Electr. Eng.* **2018**, *70*, 243–260. [CrossRef]
54. Sudholt, D. The benefits of population diversity in evolutionary algorithms: A survey of rigorous runtime analyses. In *Theory of Evolutionary Computation*; Springer: Cham, Switzerland, 2020; pp. 359–404.
55. Pisinger, D. Where are the hard knapsack problems? *Comput. Oper. Res.* **2005**, *32*, 2271–2284. [CrossRef]
56. Vasquez, M.; Yannick, V. Improved results on the 0-1 multidimensional knapsack problem. *Eur. J. Oper. Res.* **2005**, *165*, 70–81. [CrossRef]
57. Dirac, P.A.M. *The Principles of Quantum Mechanics*, 4th ed.; Oxford University Press: New York, NY, USA, 1981; p. 12.
58. Wang, J.S.; Li, S.X. An improved grey wolf optimizer based on differential evolution and elimination mechanism. *Sci. Rep.* **2019**, *9*, 1–21. [CrossRef] [PubMed]
59. Woolson, R.F. Wilcoxon signed-rank test. In *Wiley Encyclopedia of Clinical Trials, 1–3*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2007.