A Competitive Learning QUasi Affine TRansformation Evolutionary for Global Optimization and Its Application in CVRP

Nengxian Liu[1], Jeng-Shyang Pan[1,2,3], Shu-Chuan Chu[3]

[1] College of Mathematics and Computer Science, Fuzhou University, China
[2] Fujian Provincial Key Lab of Big Data Mining and Applications, Fujian University of Technology, China
[3] College of Computer Science and Engineering, Shandong University of Science and Technology, China
lylnx@fzu.edu.cn, jspan@cc.kuas.edu.tw, scchu0803@gmail.com

Abstract

In this paper, we propose a new Competitive Learning QUasi Affine TRansformation Evolutionary (CL-QUATRE) algorithm for Global Optimization and its application in Capacitated Vehicle Routing Problem (CVRP). In the proposed CL-QUATRE, the population is divided into two subpopulations (i.e., winner and loser) with a pair wise competition mechanism. Each subpopulation utilizes different mutation strategy to reserve the population diversity and improve convergence speed. The winner evolves with a mutation strategy "QUATRE/best/1", whereas the loser evolves with a modified mutation strategy "QUATRE/target-to-best-winner/1", which learns from winner subpopulation to make the algorithm more efficient. Meanwhile, a scale factor updating method, called stochastic scale factor, is introduced into the proposed CL-QUATRE algorithm to jump out of the local optima and avoid falling into stagnation. With these modifications, the proposed algorithm can achieve good balance between exploration and exploitation capability. We compare the proposed algorithm with four QUATRE variants, four DE variants, and four PSO variants on CEC2013 test suite, CEC2014 test suite and two CVRP benchmarks. The experimental results demonstrate that the CL-QUATRE algorithm achieves better or competitive performance.

Keywords: Capacitated vehicle routing problem, Competitive learning, Differential evolution, Global optimization, QUasi Affine tRansformation evolutionary algorithm

1 Introduction

In the recent decades, evolutionary computation has made great progress in both theory and practice due to great optimization demands arising from scientific and engineering community. Evolutionary computation is one of the effective methods to solve traffic problems. The idea of evolutionary computation using Darwin's principles to solve problems automatically originated in the 1950s [1]. Since then, many nature-inspired optimization approaches, including Particle Swarm Optimization (PSO) [2-4], Differential Evolution (DE) [5-8], Ant Colony Optimization (ACO) [9], Artificial Bee Colony (ABC) Optimization [10-11], Bat Algorithm (BA) [12], and QUasi-Affine TRansformation Evolution (QUATRE) algorithm [13], and so on, have been proposed to meet these demands attempting to find the optimal solutions for complex optimization problems. Among them, PSO and DE are two popular and powerful algorithms, which have attracted many researchers' attention and have been applied in various fields. In [13], Meng et al. proposed the QUATRE algorithm and discussed the relationship between QUATRE and these two algorithms. The QUATRE algorithm can also be regarded as the Parameter-reduced or an enhancement of DE algorithm [14]. Literature [14] also discussed the relationship between QUATRE algorithm and DE algorithm in more detail and demonstrated that QUATRE conquers representational or positional bias of DE algorithm. In [15], Pan gave several variants of the QUATRE algorithm. Kinds of QUATRE variants can be denoted by conventional notation "QUATRE/x/y" which is more general than the notation for DE "DE/x/y/z" [15]. The other studies related to QUATRE algorithm can be found in [16-22].

The QUATRE has been demonstrated to be a powerful algorithm according to previous studies [13-22]. It has the advantages of simplicity, less control parameters, easy implementation and convenient to be used. However, similar to other population-based algorithm, it has some drawbacks such as it may easily trap into a local optimal solution region, will be premature convergence, and will search stagnation. Therefore, avoiding the local optimal solutions and accelerating convergence speed are two critical issues for QUATRE. Maintaining population diversity is one

*Corresponding Author: Jeng-Shyang Pan; E-mail: jspan@cc.kuas.edu.tw
DOI: 10.3966/160792642020122107002

of the important approaches to alleviating aforementioned weaknesses. It is important to make a trade-off between the population diversity and convergence speed. C-QUATRE [17] has been proposed to improve the performance of QUATRE algorithm, which randomly divides the population into two groups and only evolves the individual who loses in pairwise competition. In previous literatures [23-30], we also can find the other algorithms partitioning entire population into several groups or sub-populations to reserve population diversity and to improve the performance of algorithms, such as DE [23-25], PSO [26-27], CMA-ES [28-29], and ABC [30]. In [31], Cheng proposed an improved PSO named CSO by using pairwise competition mechanism, and in CSO the loser can learn from the winner. Meanwhile, the performance of the QUATRE algorithm is significantly dependent on the mutation strategy and the control parameter scale factor F. Due to different mutation strategies having different searching capabilities and convergence speeds, different mutation strategies for the QUATRE algorithm have different performances on different optimization problems [16]. The Scaling Factor F affects the exploration and exploitation searching capability of QUATRE algorithms. Commonly, F = 0.7 is a good choice for QUATRE algorithm to solve many optimization problems [13]. The recommended effective range of F for DE is interval [0.4, 1]. Stochastic methods have been widely used to improve the efficiency of intelligent optimization algorithms, such as BA [32], PSO [33], and DE [34], because they can increase the diversity and flexibility of the population by random changing the parameters. Hence, in this paper, in order to improve the performance of QUATRE algorithm, we propose a Competitive Learning QUATRE algorithm, called CL-QUATRE, with a modified mutation strategy and a stochastic scale factor F. In the proposed CL-QUATRE algorithm, a pairwise competition mechanism is used to divide the population into two subpopulations (i.e., winner and loser). The winner and the loser have different mutation strategies. The loser can learn from the winner with a modified mutation strategy “QUATRE/ target- to-best-winner/1”.

The capacitated vehicle routing problem (CVRP) is one of the most important issues in the field of logistics, transportation and supply chain management. CVRP is a well-known NP-hard problem [35] in combinational optimization. The aim of CVRP is to determine a set of routes for a fleet of vehicles from a given depot to serve a set of customers with certain demands, and then come back to the same depot without violating the vehicle capacity constraints. Each customer should be serviced exactly once by one of the vehicles, and all customers must be assigned to vehicles [36]. In practice, the objective of this problem is to minimize the total cost of the all routes for a fleet of vehicles. Generally, the cost is measured in terms of travel distance as cost is closed associated with distance. The CVRP is a basic vehicle routing problem and it can be extended to other variants of Vehicle Routing Problem VRP, such as VRP with time windows (VRPTW).

Great progress has been made in the field of CVRP since Dantzig and Ramser proposed CVRP in 1959 [37]. The approaches developed for solving the CVRP problems can be divided into three categories, namely exact, heuristic, and meta-heuristic algorithms. In [38], Toth and Vigo reported that exact algorithm cannot consistently solve CVRP problems of more than 50 customers because the calculation consumption will grow exponentially as the difficulty of problem increasing. Therefore, heuristic and meta-heuristic algorithms have been developed for large-scale problems. In recent decades, researchers have done a lot of research on CVRP and developed various meta-heuristic algorithms such as Tabu Search (TS) [39], Simulated Annealing [40], PSO [41-43], Genetic Algorithm (GA) [44] and Symbiotic Organisms Search (SOS)[45]. The QUATRE algorithm has been proposed for solving continuous optimization problems and has a promising performance in testing complex continuous mathematical benchmark problems. However, it has not been tested for discrete problems such as CVRP. Therefore, in this paper, we apply the proposed CL-QUATRE to solve the CVRP problem and compare it with PSO, GA and SOS algorithms.

Contributions behind in this paper are as follows.

- A proposed novel CL-QUATRE algorithm reduces the deficiencies of the original QUATRE.
- A modified mutation strategy “QUATRE/target-to-best-winner/1” is proposed in which the loser can learn from the winner.
- Compared results with four QUATRE variants, four DE variants, and four PSO variants on testing CEC2013 and CEC2014 test suite are to evaluate the performance of the proposed algorithm.
- We also evaluate the performance of proposed algorithm on two CVRP benchmarks.

The rest of the paper is organized as follows. Section 2 briefly reviews the QUATRE algorithm and capacitated vehicle routing problem. The proposed CL-QUATRE algorithm and its application in CRVP are present in Section 3. The experimental analysis of the proposed algorithm under CEC2013 test suite and CEC2014 test suite for single objective real parameter optimization and CRVP problem is presented in Section 4. Finally, Section 5 draws the conclusion.

2 Related Works

2.1 Canonical QUATRE Algorithm

QUATRE [13] is a new proposed population based cooperative algorithm for solving optimization problems. QUATRE is a combination of acronyms, QUASi-Affine TRansformation Evolution. In QUATRE, individuals evolve with a quasi-affine transformation form. The exact evolution equation for QUATRE is shown as follow.

$$\mathbf{X} \leftarrow \mathbf{M} \otimes \mathbf{X} + \overline{\mathbf{M}} \otimes \mathbf{B} \qquad \textbf{(1)}$$

$\mathbf{X}$ represents the population matrix with ps individuals, $X = [\mathrm{X}_{1,\mathrm{G}}, \mathrm{X}_{2,G}, \ldots, \mathrm{X}_{\mathrm{ps},G}]^T$. $\mathrm{X}_{i,\mathrm{G}} = [\mathrm{x}_{i1}, \mathrm{x}_{i2}, \ldots, \mathrm{x}_{iD}]$ is the i^{th} row vector of the matrix $\mathbf{X}$, $\mathrm{X}_{i,\mathrm{G}}$ represents the coordinate of i^{th} individual of the G^{th} generation, and it is a candidate solution for the optimization problem, and D represents the dimension number of target function, where $i \in \{1, 2, .., \mathrm{ps}\}$. $B = [\mathrm{B}_{1,G}, \mathrm{B}_{2,G}, \ldots, \mathrm{B}_{\mathrm{ps},G}]^T$ represents the mutation matrix. "$\otimes$" represents component-wise multiplication of the elements in each matrix, which is the same as ".*" operation in Matlab software. $\mathbf{M}$ denotes an evolution matrix, and $\overline{\mathbf{M}}$ denotes a binary inverting matrix of $\mathbf{M}$. The elements of them are binary values, either 0 or 1. The binary invert operation means to invert the values of the matrix. The corresponding values of zero elements in $\mathbf{M}$ are ones in matrix $\overline{\mathbf{M}}$, while the corresponding values of one elements in $\mathbf{M}$ are zeros in $\overline{\mathbf{M}}$. Eq.2 gives an example of binary inversing operation.

$$\mathbf{M} = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & & \ldots & \\ 1 & 1 & \ldots & 1 \end{bmatrix}, \overline{\mathbf{M}} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ & & \ldots & 1 \\ 0 & 0 & \ldots & 0 \end{bmatrix} \qquad \textbf{(2)}$$

$\mathbf{M}$ is an automatically generated matrix transforming from an initial matrix $\mathbf{M}_{init}$ which is initialized according to a lower triangular matrix with all elements equalling to 1. An example of the initialization and transformation is given in Eq.3 when the population size ps equals to dimension number D. There are two consecutive steps for transforming from matrix $\mathbf{M}_{init}$ to $\mathbf{M}$, denoted by "~" operator in Eq.3. In the first step, the elements in each row vector of $\mathbf{M}_{init}$ are randomly permuted. In the second step, the sequence of the row vectors is randomly permuted without changing the elements of each row vector. When the population size ps is larger than the dimension, the matrix $\mathbf{M}_{init}$ needs to be extended in accordance with population size ps. Eq.4 gives an example of extension for $ps = 2D + 2$. More generally, when $ps\%D = k$, the top k rows of the $D \times D$ lower triangular matrix are added in $\mathbf{M}_{init}$, and adaptively modify $\mathbf{M}$ in accordance with $\mathbf{M}_{init}$ [13].

$$\mathbf{M}_{init} = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & & \ldots & \\ 1 & 1 & \ldots & 1 \end{bmatrix} \sim \begin{bmatrix} & 1 & 1 & \\ 1 & 1 & \ldots & 1 \\ & & \ldots & \\ & 1 & & \end{bmatrix} = \mathbf{M} \qquad \textbf{(3)}$$

$$\mathbf{M}_{init} = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & & \ldots & \\ 1 & 1 & \ldots & 1 \\ 1 & & & \\ 1 & 1 & & \\ & & \ldots & \\ 1 & 1 & \ldots & 1 \\ & & \ldots & \\ 1 & & & \\ 1 & 1 & & \end{bmatrix} \sim \begin{bmatrix} 1 & & \ldots & 1 \\ & 1 & \ldots & \\ & \ldots & & \\ 1 & 1 & & \\ & 1 & & \\ 1 & \ldots & 1 & 1 \\ & & 1 & \\ 1 & 1 & \ldots & 1 \\ & & \ldots & \\ 1 & \ldots & 1 & 1 \\ & & 1 & \end{bmatrix} = \mathbf{M} \qquad \textbf{(4)}$$

Literatures [14-15] have given several different mutation strategies for calculation $\mathbf{B}$ in QUATRE algorithm. They are listed in Eqs. (5)-(11). $\mathbf{X}_{gbest,G} = [\mathrm{X}_{gbest,G}, \mathrm{X}_{gbest,G}, \ldots, \mathrm{X}_{gbest,G}]^T$ is the global best matrix of G^{th} generation with each row vector equalling to the global best individual $\mathbf{X}_{gbest,G}$. Mutation scale factor F is a positive control parameter for scaling the difference matrix, whose value range is $(0,1]$ for most optimization problems. $\mathbf{X}_{ri,G}$ is a random matrix which is generated by randomly permutating the sequence of row vectors in the population matrix $\mathbf{X}$ of the G^{th} generation, where $i \in \{1, 2, .., 5\}$.

QUATRE/best/1:

$$\mathbf{B} = \mathbf{X}_{gbest,G} + F \cdot (\mathbf{X}_{r1,G} - \mathbf{X}_{r2,G}) \qquad \textbf{(5)}$$

QUATRE/rand/1:

$$\mathbf{B} = \mathbf{X}_{r1,G} + F \cdot (\mathbf{X}_{r2,G} - \mathbf{X}_{r3,G}) \qquad \textbf{(6)}$$

QUATRE/ target /1:

$$\mathbf{B} = \mathbf{X} + F \cdot (\mathbf{X}_{r1,G} - \mathbf{X}_{r2,G}) \qquad \textbf{(7)}$$

QUATRE/ target-to-best /1:

$$\mathbf{B} = \mathbf{X} + F \cdot (\mathbf{X}_{gbest,G} - \mathbf{X}) + F \cdot (\mathbf{X}_{r1,G} - \mathbf{X}_{r2,G}) \qquad \textbf{(8)}$$

QUATRE/best/2:

$$\mathbf{B} = \mathbf{X}_{gbest,G} + F \cdot (\mathbf{X}_{r1,G} - \mathbf{X}_{r2,G}) + F \cdot (\mathbf{X}_{r3,G} - \mathbf{X}_{r4,G}) \qquad \textbf{(9)}$$

QUATRE/rand/2:

$$\mathbf{B} = \mathbf{X}_{r1,G} + F \cdot (\mathbf{X}_{r2,G} - \mathbf{X}_{r3,G}) + F \cdot (\mathbf{X}_{r4,G} - \mathbf{X}_{r5,G}) \qquad \textbf{(10)}$$

QUATRE/target/2:

$$\mathbf{B} = \mathbf{X} + F \cdot (\mathbf{X}_{r1,G} - \mathbf{X}_{r2,G}) + F \cdot (\mathbf{X}_{r3,G} - \mathbf{X}_{r4,G}) \qquad \textbf{(11)}$$

2.2 Capacitated Vehicle Routing Problem

Generally, capacitated vehicle routing problem can be described as follows. Assume there is one central depot, and a fleet of vehicles dispatch goods to a set of customers from the given depot, and all the vehicles must return to the same depot after serving customers on their route. The depot and the customers' locations are given. The objective is to determine an optimal route which minimizes the travel distance or the total cost with the certain constraints. Firstly, each customer should be served exactly once by one of vehicles. Secondly, the total demand of each route does not exceed the capacity of the vehicle. Thirdly, the total length of each route does not exceed the constraint. Fourthly, the total cost is minimized. The model of CVRP is described as follows [41].

Coefficients and Decision variables

k Vehicle index, $k = 1,...,m$

i, j Customer index $i = 1,..., \mathrm{n}$, $j = 1,..., \mathrm{n}$

c_{ij} Distance/Cost from customer i to customer j

d_i Demand of customer i

Q Capacity of vehicle

D Total distance travelled by all vehicles

$$x_{ijk} = \begin{cases} 1 \text{ if vehicle depart from } i \text{ to } j \\ 0 \text{ otherwise} \end{cases} \quad \textbf{(12)}$$

$$y_{ik} = \begin{cases} 1 \text{ if vehicle } i \text{ is servedby vehicle } k \\ 0 \text{ otherwise} \end{cases} \quad \textbf{(13)}$$

Objective function

$$\text{Min } D = \sum_{k=1}^{m} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ijk} \quad \textbf{(14)}$$

Constraints

$$\sum_{i=1}^{n} x_{i0k} - \sum_{j=1}^{n} x_{0jk} = 0, \quad \forall k = 1,...,m \quad \textbf{(15)}$$

$$\sum_{i=0}^{n} \sum_{k=0}^{m} x_{ijk} = 1, \quad \forall j = 1,...,n \quad \textbf{(16)}$$

$$\sum_{j=0}^{n} \sum_{k=0}^{m} x_{ijk} = 1, \quad \forall i = 1,...,n \quad \textbf{(17)}$$

$$\sum_{j=0}^{n} x_{0jk} \le 1, \quad \forall k = 1,..., \mathrm{m} \quad \textbf{(18)}$$

$$\sum_{i=0}^{n} x_{ijk} = y_{jk}, \quad \forall j = 0,1,...,n; k = 1,..., \mathrm{m} \quad \textbf{(19)}$$

$$\sum_{j=0}^{n} x_{ijk} = y_{ik}, \quad \forall i = 0,1,...,n; k = 1,..., \mathrm{m} \quad \textbf{(20)}$$

$$\sum_{i=1}^{n} d_i y_{ik} \le Q, \quad \forall k = 1,..., \mathrm{m} \quad \textbf{(21)}$$

The objective of function Eq. (14) is to minimize total travel distance. Eq. (15) means that number of vehicles that arrives at and departs from central depot is the same. Eqs. (16) and (17) guarantee that each customer is served exactly once by one vehicle. The inequality (18) means that up to m vehicles can be used. Eqs. (19) and (20) give the relation between two decision variables. The inequality (21) ensures that the total demand of any route must not exceed the vehicle capacity.

3 CL-QUATRE and Its Application in CRVP

3.1 Competitive Learning QUATRE Algorithm (CL-QUATRE)

The main idea of our proposed CL-QUATRE algorithm is depicted in this subsection. As mentioned above, the original QUATRE algorithm suffers from the problems of losing population diversity too early in search process and easily trapping into local optima. In order to reduce above drawbacks, we introduce an improved QUATRE algorithm which consists of population initialization, population division through a pairwise competition, subpopulation evolution, subpopulation recombining and stochastic approach for changing parameter scale factor F. Figure 1 gives an illustration of the main framework of the proposed CL-QUATRE algorithm.

3.1.1 Population Division and New Mutation Strategies

Two or multi subpopulations and each subpopulation using different evolutionary operation is a significant approach to improve the performance of evolutionary algorithms [23-30], because this approach can help the evolutionary algorithms to achieve good balance between exploration and exploitation capability during search procedure, maintain population diversity and alleviate premature convergence. In [18, 26, 31], the authors used the competition mechanism to improve the efficiency of swarm-based algorithms. Inspired by these ideas, we use them to maintain population diversity and improve efficiency of QUATRE algorithm. In our proposed algorithm, we first initialize population. Then, as shown in Figure 1, the entire population is equally (assuming *ps* is an even number) divided into two subpopulations using a pairwise competition mechanism based on their fitness values, name pop_{winner} and pop_{loser} respectively. The experimental

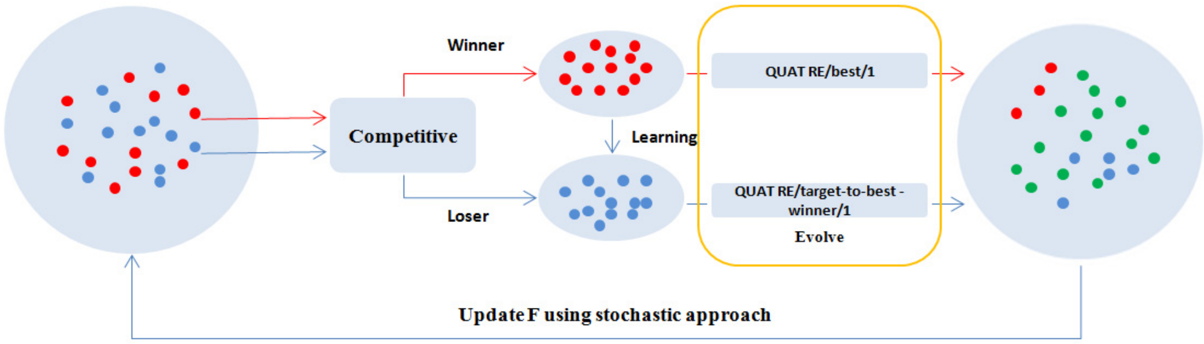

Figure 1. The main framework of CL-QUATRE

results demonstrated that different mutation strategies for QUATRE have different performance in solving various optimization problems [16]. Mutation strategy "QUATRE/best/1" has strong local search ability and fast convergence speed due to the fact that it employing the best individual from the entire population to guide the other individuals. So the subpopulation pop_{winner} evolves utilizing mutation strategy "QUATRE/best/1" to make a good exploitation around the winner individuals and to speed up the convergence. On the other hand, mutation strategy "QUATRE/target-to-best/1" yields mutation individual using the best individual and two random selected individuals so it has strong exploration ability. And the subpopulation pop_{loser} evolves adopting a modified mutation strategy based on a strategy named "QUATRE/target-to-best-winner/1" to be responsible for exploration and to maintain the population diversity. In order to make full use of the information of winner subpopulation, we modify the mutation strategy "QUATRE/target-to-best/1" to the following expression:

$$\mathbf{B} = \mathbf{X} + F \cdot (\mathbf{X}_{gbest,G} - \mathbf{X}) + F \cdot (\overline{\mathbf{X}_{r1,G}} - \mathbf{X}_{r2,G}) \quad \textbf{(22)}$$

where $\overline{\mathbf{X}_{r1,G}}$ is a random selected individual matrix from the winner subpopulation. Hence, this benefits the loser subpopulation to learn from the winner subpopulation. Therefore, this population division and each subpopulation using different mutation strategy approach in the proposed method can make a trade-off between exploration and exploitation, and thus the proposed method achieve a good balance between the convergence speed and population diversity.

3.1.2 Stochastic Scale Factor

Scale factor F plays a significant role in optimization performance of QUATRE variants. The QUATRE with different scaling factor values may be suitable for different kind of optimization problems [13]. Finding a good method to set the scale factor value is a challenging and significant task. Three types of parameter control methods have been introduced to adjust the evolutionary algorithms parameter values dynamically [46]. Among them, the stochastic technology is a simple and effective method compared with complicated parameter adaptation scheme. Therefore, we use the stochastic technology to vary the scale factor which can be expressed as follows:

$$F = \mu_{\min} + rand \cdot (\mu_{\max} - \mu_{\min}) + \sigma \cdot randn \quad \textbf{(23)}$$

where, $\mu_{\min}$ is the minimum value of the scale factor, $\mu_{\max}$ is the maximum value of the scale factor, $rand$ is a uniformly distributed random number between 0 and 1. σ is the deviation between the scale factor F and its mean, and $randn$ is a random number of standard normal distribution. This makes stochastic variations in the amplification of the difference matrix, thereby benefit to maintain the diversity of the population in the search process. In addition, this helps the individuals to jump out of the local optima and avoid stagnation, even on multimodal functions landscapes.

The pseudo code of the proposed CL-QUATRE algorithm is shown in Algorithm 1.

Algorithm 1. CL-QUATRE Algorithm

Initialization:

Initialize the solution space V, dimension D, the benchmark function $f(X)$, current generation $Gen = 1$, maximal number of generation $MaxGen$, Initialize the population $\mathbf{X}$ randomly, and calculate each individual's fitness value

Iteration:

1. **while** $Gen < MaxGen$ **do**
2. Randomly partition the population into pop_{winner} and pop_{loser} with a pairwise competition mechanism based on fitness values
   // pop_{winner} evolve
3. Generate evolution matrix $\mathbf{M}_{winner}$ using (4) and inverting matrix $\overline{\mathbf{M}}_{winner}$ using (2)
4. Calculate mutation matrix $\mathbf{B}_{winner}$ using (5)
5. Evolve sub-population pop_{winner} using (1)

6. Calculate each individual's fitness value
7. for $i = 1$ to $ps/2$ do
8. if $f(X_{i,G})$ is better than $f(X_{i,G+1})$ then
9. $X_{i,G+1} \leftarrow X_{i,G}$
10. end if
11. end for

// pop_{loser} evolve

12. Generate evolution matrix $\mathbf{M}_{loser}$ using (4) and inverting matrix $\overline{\mathbf{M}}_{loser}$ using (2)
13. Calculate mutation matrix $\mathbf{B}_{loser}$ using (22)
14. Evolve sub-population pop_{loser} using (1)
15. Calculate each individual's fitness value
16. for $i = 1$ to $ps/2$ do
17. if $f(X_{i,G})$ is better than $f(X_{i,G+1})$ then
18. $X_{i,G+1} \leftarrow X_{i,G}$
19. end if
20. end for
21. Combine pop_{winner} and pop_{loser}
22. $X_{gbset} = opt\{X_{i,G+1}\}$
23. Update F according to (23).
24. $Gen = Gen + 1$
25. end **while**

Output:

The global best solution X_{gbset}, and the best fitness value $f(X_{gbset})$.

3.2 QUATRE for Capacitated Vehicle Routing Problem

The QUATRE algorithm has been proposed to solve a continuous optimization problem and achieve great success, but the same conditions are not suitable for solving the CVRP problem. In order to apply the QUATRE algorithm to solve the combinatorial CVRP problem, continuous values need to encoding/decoding for representing solution of discrete values. In general, solving combinatorial optimization problem requires local search to enhance the quality of the solution. The flow chart of using QUATRE to solve the CVRP problem is presented in Figure 2.

3.2.1 Solution Representation

Some encoding/decoding methods are introduced in [41-43]. In [41], Wu et al. proposed to use real number coding method. Chen et al. [42] applied a quantum discrete version of PSO to CVRP. Ai and Kachitvichyanukul [43] proposed two encoding methods named SR-1 and SR-2 for CVRP, which were also used in [45]. In this study, we also use the encoding method proposed by Ai and Kachitvichyanukul because these two methods have better efficiency and performance. The solution

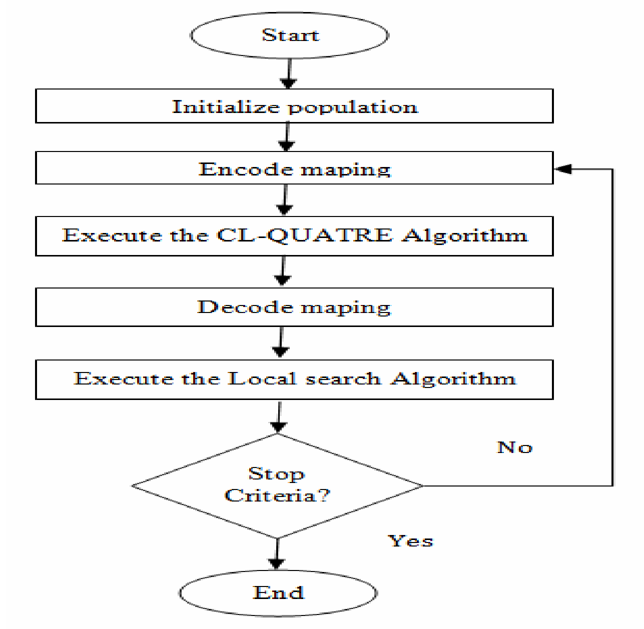

Figure 2. The flow chart by using CL-QUATRE to solve the CVRP

representation SR-1 is a $(n+2m)$ dimensional individual in which n is the number of customers and m is the number of vehicles and each dimension takes a real number. The first n dimension denotes the customer priority order and the remaining $2m$ dimension denotes the vehicle orientation. The solution representation SR-2 is a $3m$ dimensional individual in which m is the number of vehicles and all the three dimensions are used for a vehicle. The first two dimensions represent the vehicle orientation, and the last dimension represents the vehicle coverage radius. The decoding method follows the details in Ref [43]. The illustration of SR-1 and SR-2 solution representation is given in Figure 3.

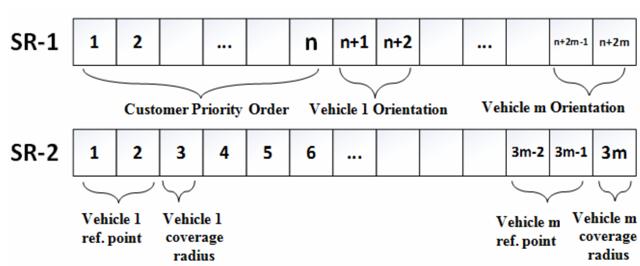

Figure 3. The illustration of SR-1 and SR-2 solution representation for CVRP

3.2.2 Fitness Function

Fitness value is used to evaluate the adaptability to environment of individual in the population. In general, choosing the appropriate object function as a fitness function to represent each individual's adaptability is one of the critical factors for successfully solving the related problems using the CL-QUATRE algorithm. According to the description in Section 2.2, the objective of CVRP is to minimize the total cost or distance with the constraints (15) – (21). When an individual violates constraints, we use a penalty function to deal with the infeasible individual. Therefore, the fitness function is defined as fellow.

$$fitness(X)=\begin{cases}\sum_{k=1}^{m}\sum_{i=0}^{n}\sum_{j=0}^{n}c_{ij}x_{ijk}, & \text{X is legal}\\ \text{V(X)}*M, & \text{otherwise}\end{cases} \quad \textbf{(24)}$$

V(X) is the value for an individual X violating of the constraints and M is a sufficiently large positive integer. The individual with the minimal fitness value will outperform other individuals.

4 Experiment Analyses

A set of experiments was conducted to evaluate the performance of the proposed algorithm CL-QUATRE and its application in capacitated vehicle routing problem.

4.1 Experimental Results for Global Optimization

In this subsection, several simulations are carried out to assess the performance of the proposed CL-QUATRE algorithm. There are two test suites CEC2013 [47] and CEC2014 [48] are adopted. The first test suite with 28 benchmark functions is proposed in the CEC2013 special session on real-parameter optimization. This benchmark test suite includes five unimodal functions (f1-f5), fifteen multi-modal functions (f6-f20) and eight composition functions (f21-f28). The second test suite with 30 benchmark functions is proposed in the CEC2014 special session on real-parameter optimization. This benchmark test suite includes three unimodal functions (f1-f3), thirteen multi-modal functions (f3-f16), six hybrid functions (f17-f22), and eight composition functions (f23-f30). More detailed descriptions of this 58 benchmark functions can be found in literature [47-48], and they are shifted to the same global best solution $O=\{o_1,o_2,...,o_D\}$. The first four part of this subsection are conducted on CEC2013 test suite. The last part of this subsection is conducted on CEC2014 test suite.

Several algorithms are used for comparison. Four QUATRE variants [13, 17] "QUATRE/target-to-best/1", "QUATRE/best/1", "C-QUATRE/target-to-best/1", and "C-QUATRE/best/1" are used for comparison in the first group because of CL-QUATRE using these two mutation strategies. Four DE variants DE [5], DE-RANDSF [34], ODE [49] and SPS-DE [50] and four PSO variants CSO [31], ccPSO [51], DNLPSO [52] and SLPSO [53] are used for comparison in the second group due to the close relationship with DE and PSO as abovementioned. The parameters of these algorithms are set according to the recommended values of the referenced papers, as shown in Table 1. We run each of them for 51 times independently, as these algorithms are stochastic algorithms that may have different results in different runs. The dimension number D of all functions is set to 30 and the allowed maximal number of function evaluation (NFE) is set to $10000*D$. All the algorithms are implemented in Matlab 2012a windows version and are run on a PC with Intel (R) Core(TM) i5-4590 CPU, 330GHz, 8 GB of RAM on window 7 Operating System.

Table 1. Parameters settings

Algorithm	Parameters settings
DE	$F=0.5, Cr=0.1, ps=100$
ODE	$F=0.5, Cr=0.1, Jr=0.3, ps=100$
DE-RANDSF	$Cr=0.9, ps=100$
SPS-DE	$F=0.7, Cr=0.5, ps=100, Q=32$
ccPSO	$c_1=c_2=2.05, K=0.7298, ps=100$
DNLPSO	$w_{\max}=0.9, w_{\min}=0.4, c_1=c_2=1.49445$ $pc\in[0.45,0.05], m=3, g=5, ps=100$
CSO	$m=100, \varphi=0$
SLPSO	$M=100, c_3=0.005, PL\in[0,1]$
QUATRE variants	$F=0.7, ps=100$
C-QUATRE variants	$F=0.7, ps=100$
CL-QUATRE	$\mu_{\max}=1, \mu_{\min}=0.4, \sigma=0.1, ps=100$

Comparison with QUATRE Variants

The best, average, and standard deviations obtained by the five QUATRE variants on 28 benchmark functions are collected in Table S1 in the supplementary file, denoted by "Best" and "Mean/Std", respectively. The comparion results are summarized in Table 2.

Table 2. Comparison results of CL-QUATRE with four QUATRE variants under CEC2013 test suite

	Best	Mean/Std
Algorithm	-/=/+	-/=/+
QUATRE/best/1	17/4/7	14/11/3
QUATRE/target-to-best/1	17/4/7	13/7/8
C-QUATRE/best/1	16/5/7	14/8/6
C-QUATRE/target-to-best/1	18/4/6	16/6/6

Symbols "-", "=" and "+" in the parentheses after the values denote "Worse Performance", "Similar Performance" and "Better Performance" respectively. Arithmetic value with the rule "the smaller the better" is used to measure the "Best" values. Wilcoxon's signed rank test at a level of significant α = 0.05 is used to measure the "Mean/Std" values. The convergence speed comparisons that employ the median value of the 51runs are presented in Figs. S1-S5 in the supplementary file. Comparison of diversity curves between CL-QUATRE and QUATRE variant on functions f2, f8, f12, and f26 are presented in Figure 4. As can be seen from Table 2, comparing with "QUATRE/best/1" algorithm, the proposed CL-QUATRE algorithm achieves 17 better performances, 4 similar performances, and 7 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 14 better performances, 11 similar performances, and 3 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with "QUATRE/target-to-best/1" algorithm, the proposed algorithm achieves 17 better performances, 4 similar performances, and 7 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 13 better performances, 7 similar performances, and 8 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with "C-QUATRE/best/1" algorithm, the proposed algorithm achieves 16 better performances, 5 similar performances, and 7 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 14 better performances, 8 similar performances, and 6 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with "C-QUATRE/target-to-best/1" algorithm, the proposed algorithm achieves 18 better performances, 4 similar performances, and 6 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 16 better performances, 6 similar performances, and 6 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Overall, CL-QUATRE achieves better performance than all the other four QUATRE and C-QUATRE variants on CEC2013 test suite. This is because the CL-QUATRE can make full use of different mutation strategies to get a better trade-off between exploration and exploitation ability and preserve population diversity, and it using stochastic strategy to change scale factor benefits to jump out of the local optima and avoid getting into stagnation. Additionally, it can be observed that QUATRE and C-QUATRE variants using different mutation strategy have different performance, owing to different mutation strategies having different searching capabilities. CL-QUATRE is different from the C-QUATRE. In each generation, the C-QUATRE only evolves the loser individuals, while the CL-QUATRE evolves the winner and loser individuals with different mutation strategies and losers can learn from the winners. That's the reason why the proposed CL-QUATRE algorithm has achieved better performance than C-QUATRE variant. Comparing with the QUATRE variant, CL-QUATRE perform well on multi-modal functions.

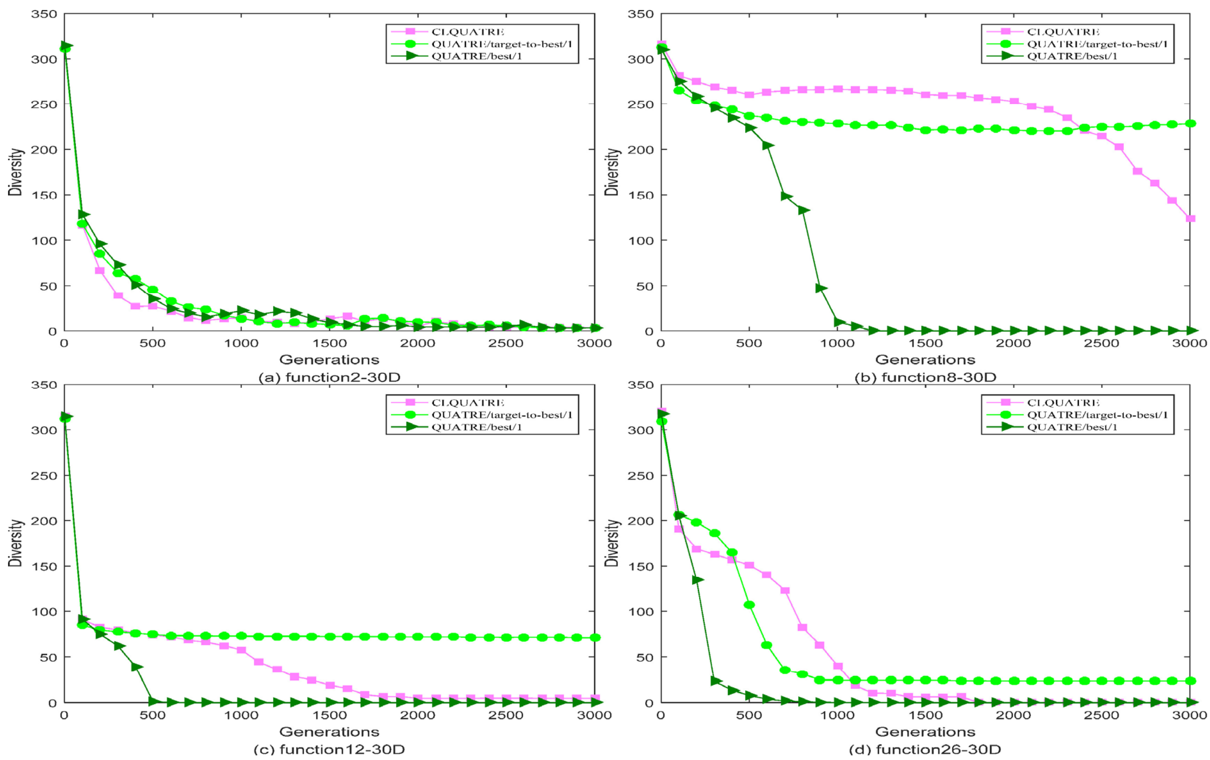

Figure 4. Comparison of diversity curves between CL-QUATRE and QUATRE on functions f2, f8, f12, and f26

From Figs. S1-S5, it can be seen that for convergence speed our proposed CL-QUATRE algorithm outperforms DE on f2-f4, f6, f9, f10, f12, f13, f15-f20, f23, f25 and f27. It also outperforms ODE on f2-f4, f6-f10, f12-f13, f15-f16, f18-f20, f23-f25, and f27. It also outperforms DE-RANDSF on f3, f7-f8, f11-f20, f22, and f24-F27. It also outperforms SPS-DE on f2, f4, f6, f8, f12, f13, f15-f18, f20, and f23. It also outperforms ccPSO on f2-f4, f6-f7, f9-f10, f12-f18, f20, and f22-f27. It also outperforms DNLPSO on f2-f4, f6, f7, f10, f12-f14, f16-f18, f20, f22-f24, and f26-f27. It also outperforms CSO on f3, f4, f6, f8, f10, f12, f13, and f16-f20. It also outperforms SLPSO on f2-f4, f6, f10, f12-f13, f15-f20, and f26. It also outperforms "QUATRE/target-to-best/1" on f9, f12-f20, f22, and f23. It also outperforms "QUATRE/best/1"on f2, f4, f6, f7, f8, f12, and f14-f20. It also outperforms "CQUATRE/target-to-best/1" on f2, f4, f6, f9, f12-f20, f22, and f23. It also outperforms "CQUATRE/ best/1"on f2, f4, f6, f8, f15-f20, f22, and f23. In summary, our proposed CL-QUATRE is competitive with other competing algorithms from convergence speed perspective. From Figure 4, for unimodal function f2 the CL-QUATRE converges to fixed point more quickly. For the other types of function, CL-QUATRE can reserve population diversity at the early stage of the evolution and converges quickly at the later stage of the search.

3.2.3 Comparison with the Other EAs

To further evaluate the performance of CL-QUATRE, we also compared it with four DE variants DE, DE-RANDSF, ODE, SPS-DE and four PSO variants CSO, ccPSO, DNLPSO and SLPSO on CEC2013 test suite. The experimental results of these compared algorithms in terms of the best, average, and standard deviations are collected in Tables S2 and S3 in the supplementary file, respectively. The comparion results are summarized in Table 3 and Table 4, respectively. Figs. S1-S5 in the supplementary file show the convergence speed of these algorithms by plotting the convergence curves of the median values of the benchmark functions.

It can be seen from Table 3 that, comparing with DE algorithm, the proposed CL-QUATRE algorithm achieves 21 better performances, 3 similar performances, and 4 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 20 better performances, 4 similar performances, and 4 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with ODE algorithm, the proposed algorithm achieves 21 better performances, 3 similar performances, and 4 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 21 better performances, 3 similar performances, and 4 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with DE-RANDSF algorithm, the proposed algorithm achieves 19 better performances, 1 similar performance, and 8 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 16 better performances, 8 similar performances, and 4 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with SPS-DE algorithm, the proposed algorithm achieves 22 better performances, 2 similar performances, and 2 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 19 better performances, 6 similar performances, and 3 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. In brief, our proposed CL-QUATRE algorithm has competitive performance in comparison with the four DE variants. Comparing with the DE variant, CL-QUATRE perform well on multi-modal functions.

Table 3. Comparison results of CL-QUATRE with four DE variants under CEC2013 test suite

	Best	Mean/Std
Algorithm	-/=/+	-/=/+
DE/best/1	21/3/4	20/4/4
ODE/best/1	21/3/4	21/3/4
DE-RANDSF/best/1	19/1/8	16/8/4
SPS-DE/best/1	24/2/2	19/6/3

Table 4. Comparison results of CL-QUATRE with four PSO variants under CEC2013 test suite

	Best	Mean/Std
Algorithm	-/=/+	-/=/+
ccPSO	24/0/4	25/3/0
dnlPSO	20/2/6	22/5/1
CSO	14/4/10	13/4/11
SLPSO	18/3/7	15/9/4

From Table 4, comparing with ccPSO algorithm, the proposed CL-QUATRE algorithm achieves 24 better performances, 0 similar performances, and 4 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 25 better performances, 3 similar performances, and 0 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with dnlPSO algorithm, the proposed algorithm achieves 20 better performances, 2 similar performances, and 6 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 22 better performances, 5 similar performances, and 1 worse performance out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with CSO algorithm, the proposed algorithm achieves 14 better performances, 4 similar performance, and 10 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 13 better performances, 4 similar performances, and 11 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Comparing with SLPSO algorithm, the proposed algorithm achieves 18 better performances, 3 similar performances, and 7 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 15 better performances, 9 similar performances, and 4 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. In summary, our proposed CL-QUATRE algorithm has competitive performance in comparison with the other four PSO variants. Comparing with the PSO variant, CL-QUATRE perform well on unimodal and multi-modal functions.

3.2.4 Effects of Stochastic Scale Factor

In order to investigate the effects of stochastic scale factor on the performance of CL-QUATRE, we compare it with CL-QUATRE using constant scale factor. In the comparison, the constant scale factor F is set to 0.7, which is the recommend value in [13]. There are three parameters ($\mu_{\min}$, $\mu_{\max}$, σ) in the Eq.23, which are set to 0.4, 1.0 and 0.1, respectively. Range [0.4, 1.0] is the effect scale factor range for QUATRE algorithm to solve most of optimization problems so that $\mu_{\min}$ is set to 0.4 and $\mu_{\max}$ is set to 1.0. Setting σ to 0.1 is obtained experimentally.

Table S4 in the supplementary file presents the experimental results of the 30-D optimization problem obtained by the CL-QUATRE algorithm with different parameter σ values. Table 5 summarizes the comparion results. As can be seen from Table 5, when $\sigma = 0.1$, CL-QUATRE seems to be good for most of benchmark functions in CEC2013. When the value of σ is set 0, 0.2, 0.3 and 0.4 respectively, the experimental results of the CL-QUATRE algorithm are relatively poor. Therefore, σ is set to 0.1 for the proposed CL-QUATRE in this paper.

Table 5. Comparison results of CL-QUATRE with different σ values on CEC2013 test suite

	Best	Mean/Std
Algorithm	-/=/+	-/=/+
CL-QUATRE $\sigma = 0$	14/3/11	2/26/0
CL-QUATRE $\sigma = 0.2$	13/4/11	8/19/1
CL-QUATRE $\sigma = 0.3$	19/2/7	15/12/1
CL-QUATRE $\sigma = 0.5$	18/3/7	17/10/1

The experimental results for scale factor on 30-D optimization problems obtained by CL-QUATRE are reported in Table S5 in the supplementary file. Table 6 summarizes the comparion results. Comparing with CL-QUATRE with const scale factor, the CL-QUATRE algorithm with stochastic scale factor achieves 16 better performances, 4 similar performances, and 8 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 11 better performances, 14 similar performances, and 3 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. The results demonstrate that CL-QUATRE with stochastic scale factor significantly outperforms CL-QUATRE with const scale factor. This is because stochastic scale factor can help the CL-QUATRE algorithm to jump out of the local optima and avoid getting into stagnation and make it more powerful. The experimental results also indicate that stochastic scale factor has significantly effect on the performance of CL-QUATRE algorithm. Therefore, we should combine the CL-QUATRE algorithm with the stochastic scale factor to get better performance.

Table 6. Comparison results of CL-QUATRE with const F = 0.7 and stochastic scale factor on CEC2013 test suite

	Best	Mean/Std
Algorithm	-/=/+	-/=/+
CL-QUATRE with const F=0.7	16/4/8	11/14/3

3.2.5 Effects of New Mutation Strategy

In order to investigate the effects of mutation strategy "QUATRE/target-to-best-winner/1" on the performance of CL-QUATRE, we compared it with mutation strategy "QUATRE/target-to-best/1" on CL-QUATRE using const scale factor F = 0.7. With fixed scale factor F value, the performance of different mutation strategies can be better observed.

Table S6 in the supplementary file reported the experimental results of the 30-D optimization problem obtained by the CL-QUATRE algorithm with fixed scale factor value and with different mutation strategies. Table 7 summarizes the comparion results. From table 7, comparing with CL-QUATRE with mutation strategy "QUATRE/target-to-best/1", the CL-QUATRE algorithm with the modified mutation strategy "QUATRE/target-to- best-winner/1" achieves 17 better performances, 3 similar performances, and 8 worse performances out of 28 benchmarks from "Best" perspective of view. It achieves 17 better performances, 8 similar performances, and 3 worse performances out of 28 benchmarks from "Mean/Std" perspective of view. Hence, the modify mutation strategy can improve the performance of CL-QUATRE algorithm and thus it is used for the proposed method in the following experiments.

Table 7. Comparison results of CL-QUATRE with fixed F = 0.7 and different mutation strategies on CEC2013 test suite

	Best	Mean/Std
Algorithm	-/=/+	-/=/+
CL-QUATRE/target-to-best/1	17/3/8	17/8/3

3.2.6 Comparing Four Competitive Algorithms on CEC2014

We further compare four competitive algorithms CSO, C-QUATRE/best/1, C-QUATRE/target-to-best/1, and CL-QUATRE on the CEC2014 test suite. Compare to CEC2013, CEC2014 benchmark problems with several novel features are more complex and make our test results more convincing.

Table S7 in the supplementary file reported the experimental results of the 30-D optimization problem obtained by four competitive algorithms. The comparion results are summarized in Table 8. From table 8, comparing with CSO algorithm, the proposed CL-QUATRE algorithm achieves 17 better performances, 2 similar performances, and 11 worse performances out of 30 benchmarks from "Best" perspective of view. It achieves 16 better performances, 2 similar performances, and 12 worse performances out of 30 benchmarks from "Mean/Std" perspective of view. Comparing with "C-QUATRE/best/1" algorithm, the proposed algorithm achieves 16 better performances, 4 similar performances, and 10 worse performances out of 30 benchmarks from "Best" perspective of view. It achieves 13 better performances, 11 similar performances, and 6 worse performances out of 30 benchmarks from "Mean/Std" perspective of view. Comparing with "C-QUATRE/target-to-best/1" algorithm, the proposed algorithm achieves 17 better performances, 4 similar performances, and 9 worse performances out of 30 benchmarks from "Best" perspective of view. It achieves 14 better performances, 6 similar performances, and 10 worse performances out of 30 benchmarks from "Mean/Std" perspective of view. Overall, the CL-QUATRE variant has better performance than the CSO, "C-QUATRE/best/1" and "C-QUATRE/target-to-best/1" algorithms.

Table 8. Comparison results of the four competitive algorithm under CEC2014 test suite

	Best	Mean/Std
Algorithm	-/=/+	-/=/+
CSO	17/2/11	16/2/12
C-QUATRE/best/1	16/4/10	13/11/6
C-QUATRE/target-to-best/1	17/4/9	14/6/10

4.2 Experimental Results for CVRP

To investigate the performance of the CL-QUATRE for solving the CVRP problem, two groups of experiments are conducted on various kinds of benchmark instances. In the first group, we compare the best value of experimental results of proposed CL-QUATRE algorithm with the results of Ref [41] using the real number solution representation. We use CL-QUATRE$_{SR\text{-}1}$ and CL-QUATRE$_{SR\text{-}2}$ to represent using CL-QUATRE to solve the CVRP problem with solution representation SR-1 and SR-2 respectively. In the second group, we compare the mean and standard deviation of the experimental results of the proposed CL-QUATRE algorithm with the results of Ref [45] using the same solution representation SR-1 and SR-2. In Ref [45], the authors use SOS to solve the CVRP problem with solution representation SR-1 and SR-2, denoted by SOS$_{SR\text{-}1}$ and SOS$_{SR\text{-}2}$ respectively. The parameters setting of CL-QUATRE are as follows. The population size is set to 50. The number of iterations is set to 1000. The other parameter is the same with the section 4.1.

In the first group, the instance is independently run 10 times because the experiment in Ref [41] was run 10 times. Table 9 shows the best value of the 10 runs. BKS means the best known solution. Dev (%) denotes the percentage of deviation from best-known solution. From the Table 9, we can see that CL-QUATRE$_{SR\text{-}2}$ has better performance than the other four algorithms. We also can see that CL-QUATRE$_{SR\text{-}1}$ has better performance than the other three algorithms, but rPSO can find three BKS values. Two visual examples of solutions for CVRP instances are shown in Figs. 5-6.

Table 9. Comparison results of the CL-QUATRE with Ref [41]

Instance	BKS	CL-QUATRE$_{SR\text{-}1}$		CL-QUATRE$_{SR\text{-}2}$		rPSO	PSO	GA
		Best	Dev(%)	Best	Dev(%)	Best	Best	Best
A-n32-k5	784	787.1	0.39	787.1	0.39	**784**	829	818
A-n33-k5	661	662.1	0.17	662.3	0.19	**661**	705	674
A-n34-k5	778	**785.2**	0.93	786.1	1.04	790	832	821
A-n39-k6	831	835.3	0.51	833.2	0.27	**831**	872	866
A-n44-k6	944	959.9	1.68	**951.2**	0.76	953	1016	991
A-n46-k7	914	918.1	0.45	**917.7**	0.41	937	977	957
A-n54-k7	1167	1187.0	1.71	**1178.8**	1.01	1202	1205	1203
A-n60-k9	1354	1370.2	1.19	**1363.3**	0.69	1407	1476	1410
A-n69-k9	1168	1187.4	1.66	**1175.6**	0.65	1231	1275	1243
A-n80-k10	1764	1830.2	3.75	**1784.1**	1.14	1864	1992	1871

In the second group, the instance is independently run 5 times because the experiment in Ref [45] was run 5 times. Table 10 shows the mean and standard deviation of the 5 runs, denoted by "Mean" and "Std", respectively. From the Table 10, we can see that when the proposed method using the solution representation SR-1, the SOS$_{SR\text{-}1}$ has better mean value than the CL-QUATRE$_{SR\text{-}1}$ and the CL-QUATRE$_{SR\text{-}1}$ has better standard deviation. We also can see that when the proposed method using the solution representation SR-2, the CL-QUATRE$_{SR\text{-}2}$ has better performance than the SOS$_{SR\text{-}2}$ in terms of mean and standard deviation.

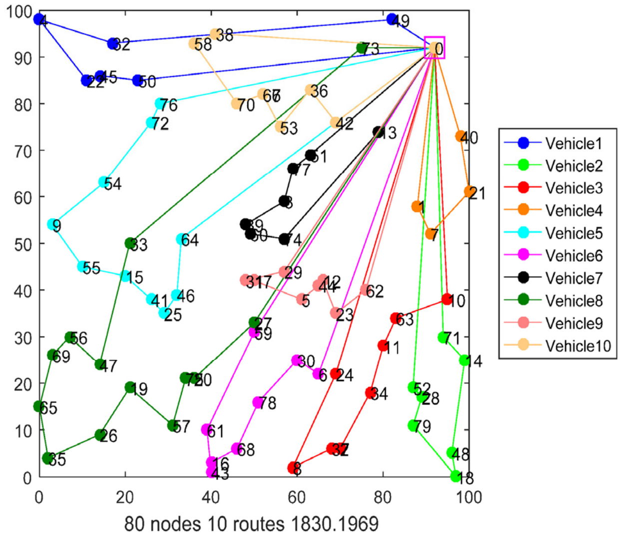

Figure 5. Visualization solution of A-n80-k10 solutions based on CL-QUATRE with SR-1

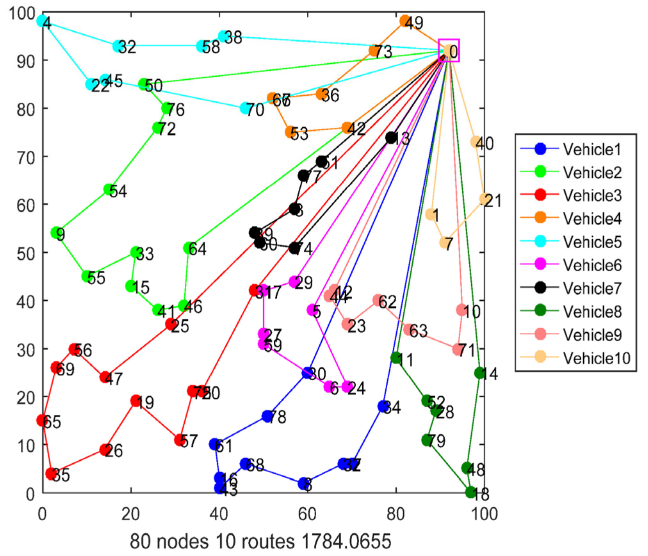

Figure 6. Visualization solution of A-n80-k10 solutions based on CL-QUATRE with SR-2

Table 10. Comparison results of the CL-QUATRE with Ref [45] on the Mean and Std

Instance	$\text{SOS}_{\text{SR-1}}$		$\text{QUATRE}_{\text{SR-1}}$		$\text{SOS}_{\text{SR-2}}$		$\text{QUATRE}_{\text{SR-2}}$	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
A-n33-k5	667.2	6.43	**664.92**	**5.20**	669.5	8.90	**664.10**	**4.10**
A-n46-k7	936.87	23.75	**921.98**	**3.79**	946.23	28.42	**919.84**	**4.62**
A-n60-k9	1401.37	20.78	**1381.32**	**7.05**	1385.93	23.20	**1363.76**	**3.65**
B-n35-k5	**956.73**	1.91	958.89	**0.00**	**957.77**	2.34	959.10	**2.23**
B-n45-k5	**753.1**	4.04	759.59	**2.17**	**753.63**	3.74	755.75	**1.01**
B-n68-k9	**1284.23**	11.27	1315.22	**1.70**	1300.8	7.74	**1293.49**	**4.12**
B-n78-k10	**1243.17**	12.05	1281.90	**8.87**	**1243.07**	12.40	1245.18	**5.42**
E-n30-k3	540.8	4.44	**539.03**	**2.36**	560.93	14.05	**544.92**	**4.39**
E-n51-k5	**542.3**	12.78	544.85	**5.06**	551.03	16.28	**528.27**	**4.07**
E-n76-k7	**715.33**	11.88	736.64	**5.13**	**713**	11.62	725.91	**6.99**

5 Conclusion

In this paper, we propose a novel CL-QUATRE algorithm for optimization problems and its application in CVRP. In the proposed CL-QUATRE, the population is partitioned into two sub-populations with a pair wise competition mechanism, and each sub-population utilizes a different mutation strategy to balance between exploration and exploitation capability and thus maintain the population diversity. Meanwhile, stochastic scale factor is introduced into the proposed CL-QUATRE algorithm to jump out of the local optima and avoid falling into stagnation. Firstly, the performance of the proposed CL-QUATRE algorithm is verified over CEC2013 test suite. The experimental results indicate that the proposed CL-QUATRE algorithm has better performance than two QUATRE variants, two C-QUATRE variants, DE, ODE, DE-RANDSF, SPS-DE, ccPSO, DNLPSO, CSO and SLPSO algorithms. Additionally, we apply the proposed algorithm to solve the CVRP problem. The experimental results show that the proposed CL-QUATRE algorithm has competitive performance than the other competing algorithms. In future, we will apply the proposed algorithm to solve more real-world optimization problems [54-56].

References

[1] S. Das, P. N. Suganthan, Differential Evolution: A Survey of the State-of-the-art, *IEEE Transactions on Evolutionary Computation*, Vol. 15, No.1, pp. 4-31, February, 2011.

[2] J. Kennedy, R. Eberhart, Particle Swarm Optimization, *IEEE International Conference on Neural Networks*, Perth, Australia, 1995, pp. 1942-1948.

[3] C. Sun, Y. Jin, J. Zeng, Y. Yu, A Two-layer Surrogate-assisted Particle Swarm Optimization Algorithm, *Soft Computing*, Vol. 19, No. 6, pp. 1461-1475, June, 2015.

[4] Y. Xue, B. Xue, M. Zhang, Self-adaptive Particle Swarm Optimization for Large-scale Feature Selection in Classification, *ACM Transactions on Knowledge Discovery from Data*, Vol. 13, No. 5, Article No. 50, October, 2019.

[5] R. Storn, K. Price, Differential Evolution- A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces, *Journal of Global Optimization*, Vol. 11, No. 4, pp. 341-359, December, 1997.

[6] Z. Meng, J.-S. Pan, HARD-DE: Hierarchical ARchive Based Mutation Strategy with Depth Information of Evolution for the Enhancement of Differential Evolution on Numerical Optimization, *IEEE Access*, Vol. 7, pp. 12832-12854, January, 2019.

[7] Z. Meng, J.-S. Pan, K.-K. Tseng, PaDE: An Enhanced Differential Evolution Algorithm with Novel Control Parameter Adaptation Schemes for Numerical Optimization, *Knowledge-Based Systems*, Vol. 168, pp. 80-99, March, 2019.

[8] Z. Meng, J.-S. Pan, L. Kong, Parameters with Adaptive Learning Mechanism (PALM) for the Enhancement of Differential Evolution, *Knowledge-Based Systems*, Vol. 141, pp. 92-112, February, 2018.

[9] M. Dorigo, V. Maniezzo, A. Colorni, Ant System: Optimization by a Colony of Cooperating Agents, *IEEE Transactions on Systems, Man, & Cybernetics Part B*, Vol. 26, No.1, pp. 29-41, February, 1996.

[10] D. Karaboga, B. Basturk, A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm, *Journal of Global Optimization*, Vol. 39, No. 3, pp. 459-471, November, 2007.

[11] Y. Xue, J. M. Jiang, B. P. Zhao, T. H. Ma, A Self-adaptive Artificial Bee Colony Algorithm Based on Global Best for Global Optimization, *Soft Computing*, Vol. 22, No. 9, pp. 2935-2952, May, 2018.

[12] X.-S. Yang, A New Metaheuristic Bat-inspired Algorithm, in: J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, N. Krasnogor (Eds.), *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, Springer, 2010, pp. 65-74.

[13] Z. Meng, J.-S. Pan, H. Xu, QUasi-Affine TRansformation Evolutionary (QUATRE) Algorithm: A Cooperative Swarm Based Algorithm for Global Optimization, *Knowledge-Based Systems*, Vol. 109, pp. 104-121, October, 2016.

[14] Z. Meng, J.-S. Pan, QUasi-affine TRansformation Evolutionary (QUATRE) Algorithm: A Parameter-reduced Differential Evolution Algorithm for Optimization Problems, *IEEE Congress on Evolutionary Computation*, Vancouver, Canada, 2016, pp. 4082-4089.

[15] J.-S. Pan, Z. Meng, H. Xu, X. Li, QUasi-Affine TRansformation Evolution (QUATRE) Algorithm: A New Simple and Accurate Structure for Global Optimization, *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, Morioka, Japan, 2016, pp. 657-667.

[16] Z. Meng, J.-S. Pan, QUasi-affine TRansformation Evolutionary (QUATRE) Algorithm: The Framework Analysis for Global Optimization and Application in Hand Gesture Segmentation, *2016 IEEE 13th International Conference on Signal Processing*, Chengdu, China, 2016, pp. 1832-1837.

[17] Z. Meng, J.-S. Pan, A Competitive QUasi-Affine TRansformation Evolutionary (C-QUATRE) Algorithm for Global Optimization, *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Budapest, Hungary, 2016, pp. 001644-001649.

[18] Z. Meng, J.-S. Pan, X. Li, The QUasi-Affine TRansformation Evolution (QUATRE) Algorithm: An Overview, *The Euro-China Conference on Intelligent Data Analysis and Applications*, Bulevar Louis Pasteur, Spain, 2017, pp. 324-333.

[19] J.-S. Pan, Z. Meng, S.-C. Chu, J. F. Roddick, QUATRE Algorithm with Sort Strategy for Global Optimization in Comparison with DE and PSO Variants, *The Euro-China Conference on Intelligent Data Analysis and Applications*, Bulevar Louis Pasteur, Spain, 2017, pp. 314-323.

[20] Z. Meng, J.-S. Pan, QUasi-Affine TRansformation Evolution with External ARchive (QUATRE-EAR): An Enhanced Structure for Differential Evolution, *Knowledge-Based Systems*, Vol. 155, pp. 35-53, September, 2018.

[21] N. Liu, J.-S. Pan, C. Sun, S.-C. Chu, An Efficent Surrogate-assisted Quasi-affine Transformation Evolutionary Algorithm for Expensive Optimization Problems, *Knowledge-Based Systems*, Vol. 209, December, 2020. DOI: 10.1016/j.knosys. 2020.106418.

[22] N. Liu, J.-S. Pan, T.-T. Nguyen, A bi-population QUasi-Affine TRansformation Evolution Algorithm for Global Optimization and its Application to Dynamic Deployment in Wireless Sensor Networks, *Eurasip Journal on Wireless Communications and Networking*, Vol. 2019, No. 1, pp. 1-12, December, 2019.

[23] W. Yu, J. Zhang, Multi-population Differential Evolution with Adaptive Parameter Control for Global Optimization, *13th Annual Genetic and Evolutionary Computation Conference,* Dublin, Ireland, 2011, pp. 1093-1098

[24] N. Lynn, R. Mallipeddi, P. N. Suganthan, Differential Evolution with Two Subpopulations, in: B. Panigrahi, P. Suganthan, S. Das (Eds.), *Swarm, Evolutionary, and Memetic Computing. SEMCCO 2014. Lecture Notes in Computer Science*, Springer, 2015, pp. 1-13.

[25] G. Wu, R. Mallipeddi, P. N. Suganthan, W. Rui, H. Chen, Differential Evolution with Multi-population Based Ensemble of Mutation Strategies, *Information Sciences*, Vol. 329, pp. 329-345, February, 2016.

[26] R. Cheng, C. Sun, Y. Jin, A Multi-swarm Evolutionary Framework Based on a Feedback Mechanism, *2013 IEEE Congress on Evolutionary Computation*, Cancun, Mexico, 2013, pp. 718-724.

[27] J.-F. Chang, S.-C. Chu, J. F. Roddick, J.-S. Pan, A Parallel Particle Swarm Optimization Algorithm with Communication Strategies, *Journal of Information Science and Engineering*, Vol. 21, No. 4, pp. 809-818, July, 2005.

[28] I. Loshchilov, M. Schoenauer, M. Sebag, Bi-Population CMA-ES Agorithms with Surrogate Models and Line Searches, *15th Annual Genetic and Evolutionary Computation Conference*, Amsterdam, The Netherlands, 2013, pp. 1177-1184.

[29] H. Chen, R. Cheng, J. Wen, H. Li, J. Weng, Solving Large-Scale Many-Objective Optimization Problems by Covariance Matrix Adaptation Evolution Strategy with Scalable Small Subpopulations, *Information Sciences*, Vol. 509, pp. 457-469, January, 2020.

[30] S. K. Nseef, S. Abdullah, A. Turky, G. Kendall, An Adaptive Multi-population Artificial Bee Colony Algorithm for

Dynamic Optimisation Problems, *Knowledge-Based Systems*, Vol. 104, pp. 14-23, July, 2016.

[31] R. Cheng, Y. Jin, A Competitive Swarm Optimizer for Large Scale Optimization, *IEEE Transactions on Cybernetics*, Vol. 45, No. 2, pp. 191-204, February, 2015.

[32] C. Gan, W. Cao, M. Wu, X. Chen, A New Bat Algorithm Based on Iterative Local Search and Stochastic Inertia Weight, *Expert Systems with Applications*, Vol. 104, pp. 202-212, August, 2018.

[33] R. C. Eberhart, Y. Shi, Particle Swarm Optimization: Developments, Applications and Resources, *IEEE International Conference on Evolutionary Computation*, Seoul, South Korea, 2001, pp. 81-86.

[34] S. Das, A. Konar, U. K. Chakraborty, Two Improved Differential Evolution Schemes for Faster Global Search, *Genetic and Evolutionary Computation Conference*, Washington, DC, USA, 2005, pp. 991-998.

[35] J. K. Lenstra, A. Kan, Complexity of Vehicle Routing and Scheduling Problems, *Networks*, Vol. 11, No. 2, pp. 221-227, Summer, 1981.

[36] A. G. Canen, N. D. Pizzolato, The Vehicle Routing Problem, *Logistics Information Management*, Vol. 7, No. 1, pp. 11-13, February, 1994.

[37] G. Dantzig, J. Ramser, The Truck Dispatching Problem, *Management Science*, Vol. 6, No. 1, pp. 80-91, October, 1959.

[38] P. Toth, D. Vigo, Eds., *The Vehicle Routing Problem, Monographs on Discrete Mathematics and Applications*, Society for Industrial & Applied, 2002.

[39] M. Gendreau, A. Hertz, G. Laporte, A Tabu Search Heuristic for the Vehicle Routing Problem, *Management Science*, Vol. 40, No. 10, pp. 1276-1290, October, 1994.

[40] A. V. Breedam, Improvement Heuristics for the Vehicle Routing Problem Based on Simulated Annealing, *European Journal of Operational Research*, Vol. 86, No. 3, pp. 480-490, November, 1995.

[41] B. Wu, W. Wang, Y. Zhao, X. Xu, F. Yang, A Novel Real Number Encoding Method of Particle Swarm Optimization for Vehicle Routing Problem, *6th World Congress on Intelligent Control and Automation*, Dalian, China, 2006, pp. 3271-3275.

[42] A. Chen, G. Yang, Z. Wu, Hybrid Discrete Particle Swarm Optimization Algorithm for Capacitated Vehicle Routing Problem, *Journal of Zhejiang University Science A*, Vol. 7, No. 4, pp. 607-614, April, 2006.

[43] T. Ai, V. Kachitvichyanukul, Particle Swarm Optimization and Two Solution Representations for Solving the Capacitated Vehicle Routing Problem, *Computers & Industrial Engineering*, Vol. 56, No.1, pp. 380-387, February, 2009.

[44] C. Ren, S. Li, New Genetic Algorithm for Capacitated Vehicle Routing Problem, in: D. Jin, S. Lin (Eds.), *Advances in Computer Science and Information Engineering*, Springer, 2012, pp. 695-700.

[45] V. F. Yu, A. A. N. P. Redi, C.-L.Yang, E. Ruskartina, B. Santosa, Symbiotic Organisms Search and Two Solution Representations for Solving the Capacitated Vehicle Routing Problem, *Applied Soft Computing*, Vol. 52, pp. 657-672, March, 2017.

[46] A. E. Eiben, R. Hinterding, Z. Michalewicz, Parameter Control in Evolutionary Algorithms, *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 2, pp. 124-141, July, 1999.

[47] J. J. Liang, B.-Y. Qu, P. N. Suganthan, A. G. Hernández-Díaz, *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-parameter Optimization*, Technical Report 201212, January, 2013.

[48] J. J. Liang, B-Y. Qu, P. N. Suganthan, *Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*, Technical Report 201311, December, 2013.

[49] S. Rahnamayan, H. Tizhoosh, M. Salama, Opposition-Based Differential Evolution, *IEEE Transactions on Evolutionary Computation*, Vol. 12, No. 1, pp. 64-79, February, 2008.

[50] S.-M. Guo, C.-C. Yang, P.-H. Hsu, J. S.-H. Tsai, Improving Differential Evolution with a Successful-Parent-Selecting Framework, *IEEE Transactions on Evolutionary Computation*, Vol. 19, No. 5, pp. 717-730, October, 2015.

[51] R. C. Eberhart, Y. Shi, Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization, *2000 Congress on Evolutionary Computation (CEC00)*, La Jolla, CA, USA, 2000, pp. 84-88.

[52] M. Nasir, S. Das, D. Maity, S. Sengupta, U. Halder, P. N. Suganthan, A Dynamic Neighborhood Learning Based Particle Swarm Optimizer for Global Numerical Optimization, *Information Sciences*, Vol. 209, No. 5, pp. 16-36, November, 2012.

[53] R. Cheng, Y. Jin, A Social Learning Particle Swarm Optimization Algorithm for Scalable Optimization, *Information Sciences*, Vol. 291, No. 6, pp. 43-60, January, 2015.

[54] J.-S. Pan, C.-Y. Lee, A. Sghaier, M. Zeghid, J. Xie, Novel Systolization of Subquadratic Space Complexity Multipliers Based on Toeplitz Matrix-Vector Product Approach, *IEEE Transactions on Very Large Scale Integration Systems*, Vol. 27, No. 7, pp. 1614-1622, July, 2019.

[55] J.-S. Pan, L. Kong, T.-W. Sung, P.-W. Tsai, Vaclav Snasel, Alpha-Fraction First Strategy for Hierarchical Model in Wireless Sensor Networks, *Journal of Internet Technology*, Vol. 19, No. 6, pp. 1717-1726, November, 2018.

[56] J. Wang, X. Gu, W. Liu, A. K. Sangaiah, H.-J. Kim, An Empower Hamilton Loop Based Data Collection Algorithm with Mobile Agent for WSNs, *Human-centric Computing and Information Sciences*, Vol. 9, No. 18, pp. 1-14, December, 2019.

Biographies

Nengxian Liu is a Ph.D. student in Fuzhou University. His current research interests include evolutionary algorithms and their applications in real-world problems.

Jeng-Shyang Pan received the Ph.D. degree in Electrical Engineering from the University of Edinburgh, U.K. in 1996. Currently, he is a Professor in Fujian University of Technology and Shandong University of Science and Technology. His current research interests include soft computing, information security and signal processing.

Shu-Chuan Chu received the Ph.D. degree in 2004 from the School of Computer Science, Engineering and Mathematics, Flinders University of South Australia. Currently, She is the Research Fellow with PhD advisor in the Shandong University of Science and Technology. Her research interests are mainly in Swarm Intelligence, Intelligent Computing and Data Mining.

Appendix

Table S1. Comparison results of the five QUATRE variants on CEC2013 test suite

30D	QUATRE/best/1		QUATRE/target-to-best/1		C-QUATRE/best/1		C-QUATRE/target-to-best/1		CLQUATRE	
	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std
1	0.000E+00(=)	3.567E-14/8.351E-14(-)	0.000E+00(=)	1.338E-14/5.403E-14(=)	0.000E+00(=)	0.000E+00/0.000E+00(=)	0.000E+00(=)	0.000E+00/0.000E+00(=)	0.000E+00	0.000E+00/0.000E+00
2	7.870E+04(-)	2.921E+05/1.679E+05(=)	6.563E+04(+)	2.511E+05/1.596E+05(=)	1.355E+05(-)	5.024E+05/2.524E+05(-)	9.032E+04(-)	3.581E+05/1.639E+05(-)	7.130E+04	2.567E+05/1.419E+05
3	5.741E-02(+)	2.710E+06/8.384E+06(=)	6.708E-10(+)	5.664E+04/2.518E+05(+)	1.553E-03(+)	1.097E+06/3.505E+06(+)	9.276E-05(+)	4.741E+04/2.245E+05(+)	1.723E-01	1.152E+06/2.325E+06
4	2.635E+00(+)	1.633E+01/1.132E+01(=)	1.211E+00(+)	8.595E+00/5.719E+00(+)	3.056E+01(-)	9.115E+01/4.483E+01(-)	5.201E+00(-)	3.863E+01/2.416E+01(-)	2.884E+00	1.553E+01/1.063E+01
5	0.000E+00(=)	1.092E-13/2.229E-14(=)	1.137E-13(-)	1.137E-13/0.000E+00(=)	0.000E+00(=)	8.917E-14/4.722E-14(+)	0.000E+00(=)	7.802E-14/5.328E-14(+)	0.000E+00	1.092E-13/2.229E-14
6	7.749E-07(-)	4.335E+00/8.190E+00(=)	1.620E-08(-)	4.317E+00/9.625E+00(+)	2.452E-05(-)	8.989E+00/8.326E+00(-)	1.226E-05(-)	5.976E+00/6.366E+00(-)	4.850E-09	4.088E+00/7.516E+00
7	1.041E+00(+)	1.799E+01/1.239E+01(=)	1.420E-01(+)	3.999E+00/4.235E+00(+)	1.280E+00(+)	1.108E+01/8.543E+00(=)	2.576E-02(+)	2.927E+00/2.849E+00(+)	1.802E+00	1.583E+01/1.557E+01
8	2.084E+01(-)	2.101E+01/5.147E-02(-)	2.083E+01(-)	2.094E+01/4.841E-02(=)	2.089E+01(-)	2.100E+01/4.420E-02(-)	2.082E+01(-)	2.094E+01/5.565E-02(=)	2.075E+01	2.094E+01/5.935E-02
9	7.762E+00(+)	1.656E+01/5.852E+00(+)	1.158E+01(-)	2.711E+01/5.835E+00(-)	7.127E+00(+)	1.716E+01/6.140E+00(+)	7.466E+00(+)	3.308E+01/6.374E+00(-)	9.803E+00	1.955E+01/4.456E+00
10	0.000E+00(=)	2.767E-02/1.582E-02(+)	0.000E+00(=)	2.695E-02/1.653E-02(+)	0.000E+00(=)	2.879E-02/1.575E-02(=)	7.396E-03(-)	4.049E-02/2.336E-02(=)	0.000E+00	3.839E-02/2.471E-02
11	1.194E+01(-)	2.535E+01/7.748E+00(-)	2.012E+01(-)	2.883E+01/2.388E+00(-)	7.960E+00(-)	1.909E+01/5.999E+00(-)	1.499E+01(-)	2.643E+01/3.580E+00(-)	4.975E+00	1.317E+01/5.218E+00
12	4.124E+01(-)	7.986E+01/2.408E+01(-)	7.722E+01(-)	1.195E+02/1.327E+01(-)	2.592E+01(-)	8.379E+01/2.552E+01(-)	1.324E+02(-)	1.538E+02/1.091E+01(-)	2.288E+01	5.057E+01/1.294E+01
13	3.519E+01(+)	1.164E+02/3.270E+01(=)	9.019E+01(-)	1.336E+02/1.579E+01(-)	6.204E+01(-)	1.111E+02/2.592E+01(=)	1.344E+02(-)	1.664E+02/9.801E+00(-)	5.957E+01	1.085E+02/2.390E+01
14	3.234E+02(-)	8.238E+02/3.089E+02(-)	9.005E+02(-)	1.363E+03/2.126E+02(-)	1.317E+02(+)	5.976E+02/2.534E+02(=)	1.082E+03(-)	1.482E+03/1.917E+02(-)	1.652E+02	5.951E+02/2.214E+02
15	3.595E+03(-)	5.296E+03/7.368E+02(-)	5.361E+03(-)	6.222E+03/3.073E+02(-)	3.667E+03(-)	5.485E+03/8.094E+02(-)	6.206E+03(-)	6.910E+03/2.555E+02(-)	2.069E+03	3.718E+03/6.164E+02
16	1.202E+00(-)	2.251E+00/4.682E-01(-)	1.098E+00(-)	2.263E+00/3.292E-01(-)	1.558E+00(-)	2.393E+00/4.272E-01(-)	1.913E+00(-)	2.439E+00/2.403E-01(-)	2.879E-01	1.144E+00/5.438E-01
17	2.511E+01(+)	5.461E+01/8.946E+00(-)	5.350E+01(-)	6.105E+01/2.794E+00(-)	3.768E+01(-)	5.059E+01/5.090E+00(-)	5.534E+01(-)	6.472E+01/2.911E+00(-)	3.398E+01	4.122E+01/4.059E+00
18	7.818E+01(-)	1.657E+02/2.786E+01(-)	1.588E+02(-)	1.914E+02/1.068E+01(-)	1.338E+02(-)	1.811E+02/2.193E+01(-)	1.736E+02(-)	2.046E+02/1.050E+01(-)	4.374E+01	7.789E+01/1.604E+01
19	1.929E+00(-)	3.597E+00/8.967E-01(-)	3.627E+00(-)	4.724E+00/4.334E-01(-)	2.149E+00(-)	3.381E+00/7.278E-01(-)	5.067E+00(-)	5.810E+00/3.113E-01(-)	1.158E+00	2.331E+00/6.108E-01
20	1.003E+01(-)	1.173E+01/6.416E-01(-)	1.054E+01(-)	1.183E+01/3.855E-01(-)	1.044E+01(-)	1.176E+01/5.963E-01(-)	1.128E+01(-)	1.210E+01/2.950E-01(-)	9.020E+00	1.088E+01/6.988E-01
21	2.000E+02(=)	2.900E+02/8.490E+01(=)	2.000E+02(=)	3.055E+02/8.428E+01(=)	2.000E+02(=)	3.046E+02/8.062E+01(=)	2.000E+02(=)	3.055E+02/8.428E+01(=)	2.000E+02	3.055E+02/8.428E+01
22	2.290E+02(-)	8.873E+02/3.233E+02(-)	1.152E+03(-)	1.484E+03/1.707E+02(-)	1.218E+02(+)	6.984E+02/2.326E+02(=)	8.384E+02(-)	1.769E+03/3.444E+02(-)	2.085E+02	6.215E+02/2.451E+02
23	4.076E+03(-)	5.196E+03/6.588E+02(-)	5.378E+03(-)	6.089E+03/3.617E+02(-)	3.366E+03(-)	5.488E+03/7.218E+02(-)	6.131E+03(-)	6.887E+03/2.664E+02(-)	2.294E+03	4.024E+03/6.851E+02
24	2.052E+02(-)	2.371E+02/1.481E+01(=)	2.002E+02(+)	2.183E+02/1.436E+01(+)	2.006E+02(+)	2.281E+02/1.282E+01(+)	2.000E+02(+)	2.095E+02/1.056E+01(+)	2.042E+02	2.374E+02/1.302E+01
25	2.461E+02(-)	2.584E+02/7.836E+00(=)	2.383E+02(-)	2.519E+02/7.672E+00(+)	2.371E+02(-)	2.509E+02/6.656E+00(+)	2.332E+02(+)	2.475E+02/8.698E+00(+)	2.339E+02	2.573E+02/9.878E+00
26	2.000E+02(-)	2.467E+02/6.396E+01(-)	2.000E+02(+)	2.227E+02/5.021E+01(=)	2.000E+02(-)	2.531E+02/6.427E+01(-)	2.000E+02(-)	2.300E+02/5.940E+01(=)	2.000E+02	2.263E+02/5.388E+01
27	4.125E+02(-)	6.638E+02/1.170E+02(+)	3.092E+02(+)	6.130E+02/1.384E+02(+)	3.410E+02(+)	6.335E+02/1.157E+02(+)	3.003E+02(+)	6.229E+02/2.640E+02(+)	4.040E+02	7.156E+02/1.371E+02
28	1.000E+02(+)	2.961E+02/2.801E+01(=)	3.000E+02(=)	3.398E+02/1.991E+02(=)	3.000E+02(=)	3.202E+02/1.445E+02(=)	3.000E+02(=)	3.199E+02/1.423E+02(=)	3.000E+02	3.422E+02/2.109E+02
-/=/+	17/4/7	14/11/3	17/4/7	13/7/8	16/5/7	14/8/6	18/4/6	16/6/6	-/-/-	-/-/-

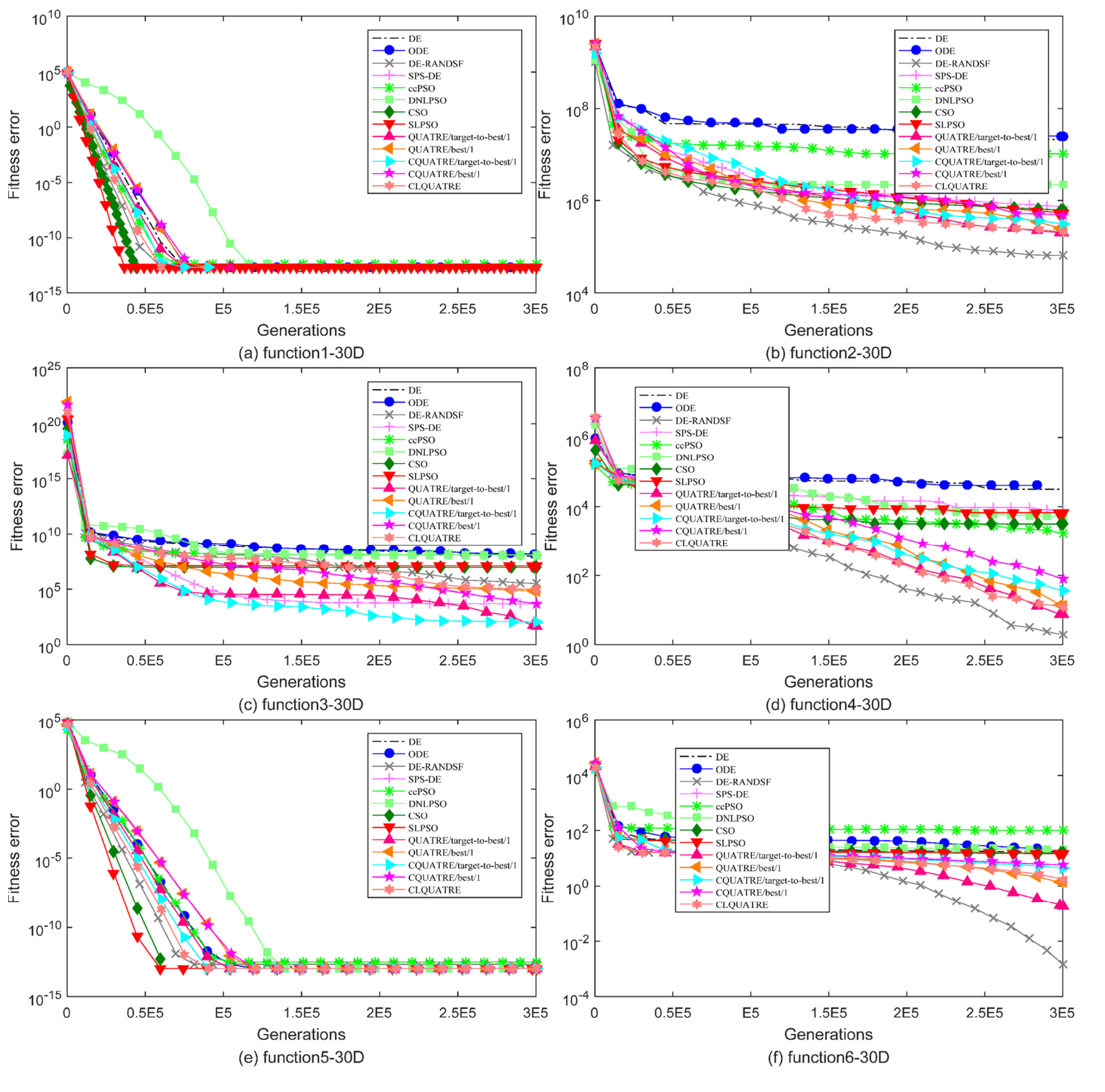

Figure S1. Comparison of the median of fitness errors for functions $f_1 - f_6$ with 30D optimization

Table S2. Comparison results of DE, ODE, DE-RANDSF, SPS-DE and CL-QUATRE algorithms on CEC2013 test suite

30D	DE/best/1		ODE/best/1		DE-RANDSF/best/1		SPS-DE/best/1		CLQUATRE	
	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std
1	0.000E+00(=)	2.185E-13/4.457E-14(-)	0.000E+00(=)	1.159E-13/1.148E-13(-)	2.274E-13(-)	2.318E-13/3.184E-14(-)	9.439E+02(-)	1.272E+03/1.113E+02(-)	0.000E+00	0.000E+00/0.000E+00
2	6.948E+06(-)	2.098E+07/7.009E+06(-)	1.075E+07(-)	2.534E+07/8.294E+06(-)	5.959E+03(+)	7.351E+04/3.776E+04(+)	2.195E+05(-)	8.046E+05/3.599E+05(-)	7.130E+04	2.567E+05/1.419E+05
3	1.149E+07(-)	1.218E+08/1.635E+08(-)	2.071E+07(-)	2.140E+08/2.288E+08(-)	2.043E+00(-)	1.461E+06/2.372E+06(=)	1.003E+03(-)	1.771E+06/4.474E+06(=)	1.723E-01	1.152E+06/2.325E+06
4	1.449E+04(-)	3.102E+04/5.984E+03(-)	2.556E+04(-)	3.931E+04/7.902E+03(-)	4.811E-02(+)	4.260E+00/7.040E+00(+)	4.554E+03(-)	7.738E+03/1.756E+03(-)	2.884E+00	1.553E+01/1.063E+01
5	1.137E-13(-)	1.137E-13/0.000E+00(=)	1.137E-13(-)	1.137E-13/0.000E+00(=)	1.137E-13(-)	2.452E-13/1.076E-13(-)	5.890E+02(-)	8.880E+02/9.098E+01(-)	0.000E+00	1.092E-13/2.229E-14
6	1.585E+01(-)	2.794E+01/1.907E+01(-)	1.556E+01(-)	2.889E+01/1.991E+01(-)	8.686E-11(+)	5.862E+00/1.091E+01(+)	5.415E+02(-)	7.936E+02/9.140E+01(-)	4.850E-09	4.088E+00/7.516E+00
7	3.026E+01(-)	4.731E+01/1.047E+01(-)	3.131E+01(-)	4.788E+01/1.021E+01(-)	4.018E+00(-)	3.218E+01/1.798E+01(-)	2.934E+02(-)	5.285E+02/1.500E+02(-)	1.802E+00	1.583E+01/1.557E+01
8	2.081E+01(-)	2.094E+01/4.724E-02(=)	2.085E+01(-)	2.097E+01/3.599E-02(-)	2.077E+01(-)	2.094E+01/5.207E-02(=)	2.118E+01(-)	2.133E+01/6.821E-02(-)	2.075E+01	2.094E+01/5.935E-02
9	2.401E+01(-)	2.928E+01/2.027E+00(-)	2.016E+01(-)	2.626E+01/3.349E+00(-)	1.016E+01(-)	1.524E+01/2.545E+00(+)	4.276E+01(-)	4.848E+01/2.264E+00(-)	9.803E+00	1.955E+01/4.456E+00
10	1.510E+00(-)	2.642E+00/7.861E-01(-)	1.802E+00(-)	4.059E+00/1.840E+00(-)	7.396E-03(-)	4.769E-02/3.130E-02(=)	2.519E+02(-)	4.529E+02/4.559E+01(-)	0.000E+00	3.839E-02/2.471E-02
11	0.000E+00(+)	4.487E-01/6.693E-01(+)	0.000E+00(+)	3.121E-01/6.130E-01(+)	3.582E+01(-)	6.266E+01/1.740E+01(-)	3.117E+02(-)	3.812E+02/1.820E+01(-)	4.975E+00	1.317E+01/5.218E+00
12	9.030E+01(-)	1.231E+02/1.734E+01(-)	7.126E+01(-)	1.120E+02/1.929E+01(-)	3.681E+01(-)	7.432E+01/2.098E+01(-)	2.638E+02(-)	2.889E+02/8.883E+00(-)	2.288E+01	5.057E+01/1.294E+01
13	1.075E+02(-)	1.400E+02/1.410E+01(-)	7.338E+01(-)	1.271E+02/2.073E+01(-)	4.339E+01(+)	1.322E+02/4.116E+01(-)	1.545E+02(-)	1.911E+02/9.357E+00(-)	5.957E+01	1.085E+02/2.390E+01
14	1.451E+00(+)	5.562E+01/7.836E+01(+)	1.472E+00(+)	2.764E+01/4.556E+01(+)	5.910E+02(-)	1.473E+03/4.122E+02(-)	1.726E+02(-)	5.770E+02/2.263E+02(=)	1.652E+02	5.951E+02/2.214E+02
15	5.207E+03(-)	6.088E+03/3.764E+02(-)	3.587E+03(-)	5.296E+03/6.841E+02(-)	2.413E+03(-)	4.202E+03/1.370E+03(=)	6.417E+03(-)	7.085E+03/3.255E+02(-)	2.069E+03	3.718E+03/6.164E+02
16	1.183E+00(-)	2.317E+00/3.327E-01(-)	1.857E+00(-)	2.390E+00/2.882E-01(-)	1.718E-01(+)	2.400E+00/4.381E-01(-)	1.517E+00(-)	2.394E+00/3.252E-01(-)	2.879E-01	1.144E+00/5.438E-01
17	3.043E+01(+)	3.048E+01/9.298E-02(+)	3.043E+01(+)	3.052E+01/1.421E-01(+)	3.278E+01(+)	8.661E+01/1.782E+01(-)	4.336E+01(-)	5.173E+01/5.944E+00(-)	3.398E+01	4.122E+01/4.059E+00
18	1.757E+02(-)	2.025E+02/1.255E+01(-)	1.528E+02(-)	1.886E+02/1.414E+01(-)	7.267E+01(-)	1.647E+02/7.096E+01(-)	1.628E+02(-)	2.002E+02/1.295E+01(-)	4.374E+01	7.789E+01/1.604E+01
19	2.240E+00(-)	3.055E+00/3.822E-01(-)	2.688E+00(-)	3.700E+00/3.298E-01(-)	2.662E+00(-)	4.461E+00/1.801E+00(-)	1.264E+00(-)	2.568E+00/1.473E+00(=)	1.158E+00	2.331E+00/6.108E-01
20	1.177E+01(-)	1.260E+01/3.226E-01(-)	1.175E+01(-)	1.267E+01/3.709E-01(-)	9.885E+00(-)	1.196E+01/8.644E-01(-)	1.073E+01(-)	1.169E+01/4.388E-01(-)	9.020E+00	1.088E+01/6.988E-01
21	2.000E+02(=)	2.996E+02/8.794E+01(=)	2.000E+02(=)	3.083E+02/8.646E+01(=)	2.000E+02(=)	3.001E+02/6.989E+01(=)	2.000E+02(=)	2.847E+02/6.943E+01(=)	2.000E+02	3.055E+02/8.428E+01
22	2.216E+01(+)	1.601E+02/8.337E+01(+)	2.077E+01(+)	2.220E+02/1.998E+02(+)	7.536E+02(-)	1.706E+03/5.378E+02(-)	2.114E+02(-)	5.739E+02/1.851E+02(=)	2.085E+02	6.215E+02/2.451E+02
23	5.174E+03(-)	6.346E+03/4.130E+02(-)	2.538E+03(-)	5.499E+03/9.217E+02(-)	2.539E+03(-)	4.222E+03/1.114E+03(=)	6.182E+03(-)	7.109E+03/3.287E+02(-)	2.294E+03	4.024E+03/6.851E+02
24	2.423E+02(-)	2.643E+02/1.057E+01(-)	2.495E+02(-)	2.647E+02/9.134E+00(-)	2.301E+02(-)	2.506E+02/9.273E+00(-)	2.000E+02(+)	2.250E+02/1.159E+01(+)	2.042E+02	2.374E+02/1.302E+01
25	2.739E+02(-)	2.866E+02/4.812E+00(-)	2.683E+02(-)	2.867E+02/6.064E+00(-)	2.514E+02(-)	2.714E+02/8.744E+00(-)	2.376E+02(-)	2.484E+02/5.420E+00(+)	2.339E+02	2.573E+02/9.878E+00
26	2.006E+02(-)	2.014E+02/4.153E-01(-)	2.007E+02(-)	2.014E+02/4.141E-01(-)	2.000E+02(+)	3.022E+02/6.379E+01(-)	2.000E+02(-)	2.567E+02/6.089E+01(-)	2.000E+02	2.263E+02/5.388E+01
27	8.814E+02(-)	1.022E+03/5.504E+01(-)	8.990E+02(-)	1.035E+03/3.968E+01(-)	4.928E+02(-)	7.463E+02/9.201E+01(=)	3.001E+02(+)	5.488E+02/8.881E+01(+)	4.040E+02	7.156E+02/1.371E+02
28	3.000E+02(=)	3.460E+02/2.301E+02(=)	3.000E+02(=)	3.000E+02/3.028E-13(=)	1.000E+02(+)	3.538E+02/2.904E+02(=)	3.000E+02(=)	3.000E+02/2.209E-13(=)	3.000E+02	3.422E+02/2.109E+02
-/=/+	21/3/4	20/4/4	21/3/4	21/3/4	19/1/8	16/8/4	24/2/2	19/6/3	-/-/-	-/-/-

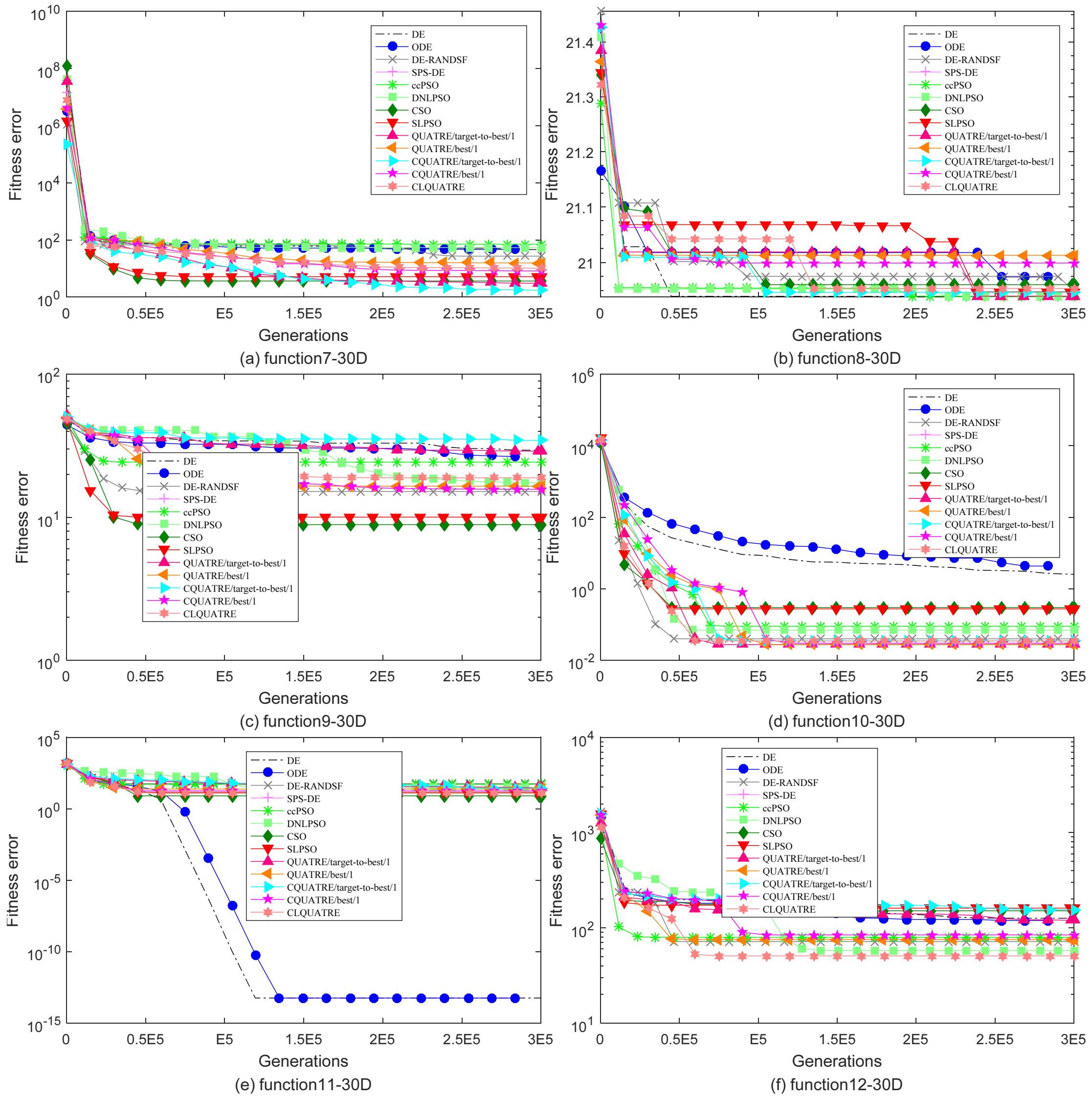

Figure S2. Comparison of the median of fitness errors for functions $f_7 - f_{12}$ with 30D optimization

Table S3. Comparison results of ccPSO, DNLPSO, CSO, SLPSO and CL-QUATRE algorithms on CEC2013 test suite

30D	ccPSO		dnlPSO		CSO		SLPSO		CLQUATRE	
	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std
1	2.274E-13(-)	3.879E-13/1.459E-13(-)	0.000E+00(=)	1.460E+01/1.020E+02(-)	0.000E+00(=)	0.000E+00/0.000E+00(=)	0.000E+00(=)	1.739E-13/9.741E-14(-)	0.000E+00	0.000E+00/0.000E+00
2	1.448E+06(-)	1.302E+07/9.058E+06(-)	2.207E+05(-)	3.568E+06/4.994E+06(-)	1.934E+05(-)	6.605E+05/3.045E+05(-)	1.102E+05(-)	5.922E+05/2.689E+05(-)	7.130E+04	2.567E+05/1.419E+05
3	8.487E+04(-)	4.736E+08/1.419E+09(-)	6.519E+03(-)	1.049E+09/2.228E+09(-)	3.450E+03(-)	1.337E+07/1.436E+07(-)	2.843E+02(-)	1.619E+07/1.439E+07(-)	1.723E-01	1.152E+06/2.325E+06
4	6.904E+02(-)	1.713E+03/6.479E+02(-)	7.809E+02(-)	1.158E+04/1.628E+04(-)	1.570E+03(-)	3.215E+03/8.860E+02(-)	2.922E+03(-)	6.813E+03/2.382E+03(-)	2.884E+00	1.553E+01/1.063E+01
5	2.274E-13(-)	4.013E-13/1.211E-13(-)	0.000E+00(=)	6.684E+00/3.026E+01(=)	0.000E+00(=)	0.000E+00/0.000E+00(+)	1.137E-13(-)	1.159E-13/1.592E-14(=)	0.000E+00	1.092E-13/2.229E-14
6	1.021E+00(-)	9.029E+01/3.804E+01(-)	2.749E-01(-)	2.767E+01/2.405E+01(-)	1.416E+01(-)	1.581E+01/3.154E+00(-)	1.363E+01(-)	1.786E+01/8.905E+00(-)	4.850E-09	4.088E+00/7.516E+00
7	2.504E+01(-)	7.333E+01/2.964E+01(-)	6.417E+00(-)	1.022E+02/1.239E+02(-)	7.361E-01(+)	4.909E+00/3.634E+00(+)	1.377E-01(+)	5.947E+00/3.846E+00(+)	1.802E+00	1.583E+01/1.557E+01
8	2.074E+01(+)	2.092E+01/5.539E-02(=)	2.076E+01(-)	2.093E+01/5.680E-02(=)	2.076E+01(-)	2.096E+01/5.056E-02(=)	2.078E+01(-)	2.094E+01/5.048E-02(=)	2.075E+01	2.094E+01/5.935E-02
9	1.423E+01(-)	2.443E+01/3.627E+00(-)	9.511E+00(+)	2.032E+01/7.952E+00(=)	5.430E+00(+)	9.169E+00/1.755E+00(+)	4.971E+00(+)	1.022E+01/2.236E+00(+)	9.803E+00	1.955E+01/4.456E+00
10	1.972E-02(-)	9.564E-02/5.006E-02(-)	7.396E-03(-)	7.680E+00/5.037E+01(-)	8.616E-02(-)	3.336E-01/1.663E-01(-)	6.160E-02(-)	2.900E-01/1.463E-01(-)	0.000E+00	3.839E-02/2.471E-02
11	1.990E+01(-)	5.092E+01/1.635E+01(-)	1.079E+01(-)	3.861E+01/2.590E+01(-)	2.985E+00(+)	7.726E+00/2.093E+00(+)	7.960E+00(-)	1.522E+01/5.315E+00(-)	4.975E+00	1.317E+01/5.218E+00
12	4.079E+01(-)	8.953E+01/3.252E+01(-)	1.990E+01(+)	6.765E+01/4.414E+01(=)	1.282E+02(-)	1.503E+02/8.689E+00(-)	1.261E+02(-)	1.588E+02/9.862E+00(-)	2.288E+01	5.057E+01/1.294E+01
13	7.459E+01(-)	1.712E+02/4.582E+01(-)	3.156E+01(+)	1.449E+02/7.822E+01(-)	1.378E+01(+)	1.496E+02/2.082E+01(-)	1.274E+02(-)	1.599E+02/1.004E+01(-)	5.957E+01	1.085E+02/2.390E+01
14	4.460E+02(-)	1.653E+03/4.278E+02(-)	5.607E+02(-)	3.108E+03/2.124E+03(-)	3.081E+01(+)	3.662E+02/1.514E+02(+)	1.866E+02(-)	6.892E+02/2.528E+02(=)	1.652E+02	5.951E+02/2.214E+02
15	2.857E+03(-)	4.258E+03/7.436E+02(-)	2.594E+03(-)	5.460E+03/1.730E+03(-)	1.570E+02(+)	1.299E+03/1.874E+03(+)	5.729E+02(+)	4.575E+03/2.514E+03(=)	2.069E+03	3.718E+03/6.164E+02
16	7.150E-01(-)	1.862E+00/5.292E-01(-)	3.446E-01(-)	2.167E+00/6.883E-01(-)	1.746E+00(-)	2.449E+00/3.252E-01(-)	1.716E+00(-)	2.445E+00/3.055E-01(-)	2.879E-01	1.144E+00/5.438E-01
17	6.105E+01(-)	8.682E+01/1.330E+01(-)	4.685E+01(-)	1.149E+02/5.231E+01(-)	1.083E+02(-)	1.319E+02/1.210E+01(-)	1.200E+02(-)	1.633E+02/1.353E+01(-)	3.398E+01	4.122E+01/4.059E+00
18	5.190E+01(-)	1.155E+02/3.394E+01(-)	7.172E+01(-)	1.553E+02/5.597E+01(-)	1.673E+02(-)	1.852E+02/8.018E+00(-)	1.649E+02(-)	1.929E+02/8.809E+00(-)	4.374E+01	7.789E+01/1.604E+01
19	1.987E+00(-)	4.101E+00/1.278E+00(-)	1.818E+00(-)	4.494E+00/2.520E+00(-)	2.487E+00(-)	3.312E+00/4.270E-01(-)	1.858E+00(-)	3.442E+00/6.180E-01(-)	1.158E+00	2.331E+00/6.108E-01
20	1.026E+01(-)	1.408E+01/1.491E+00(-)	1.170E+01(-)	1.444E+01/9.185E-01(-)	1.042E+01(-)	1.116E+01/3.655E-01(-)	1.087E+01(-)	1.347E+01/1.436E+00(-)	9.020E+00	1.088E+01/6.988E-01
21	1.000E+02(+)	3.055E+02/8.662E+01(=)	1.000E+02(+)	3.058E+02/8.047E+01(=)	2.000E+02(=)	3.150E+02/8.560E+01(=)	2.000E+02(=)	2.875E+02/7.289E+01(=)	2.000E+02	3.055E+02/8.428E+01
22	1.085E+03(-)	1.912E+03/4.176E+02(-)	9.566E+02(-)	3.113E+03/1.869E+03(-)	1.352E+02(+)	2.549E+02/1.237E+02(+)	1.493E+02(+)	5.981E+02/2.937E+02(=)	2.085E+02	6.215E+02/2.451E+02
23	2.151E+03(+)	4.490E+03/8.463E+02(-)	2.579E+03(-)	5.248E+03/1.710E+03(-)	1.104E+02(+)	6.396E+02/3.074E+02(+)	3.711E+02(+)	3.623E+03/2.628E+03(=)	2.294E+03	4.024E+03/6.851E+02
24	2.454E+02(-)	2.782E+02/1.165E+01(-)	2.287E+02(-)	2.499E+02/1.582E+01(-)	2.001E+02(+)	2.133E+02/8.364E+00(+)	2.000E+02(+)	2.225E+02/1.184E+01(+)	2.042E+02	2.374E+02/1.302E+01
25	2.716E+02(-)	2.962E+02/1.216E+01(-)	2.260E+02(+)	2.541E+02/2.131E+01(+)	2.424E+02(-)	2.517E+02/5.329E+00(+)	2.362E+02(-)	2.538E+02/7.315E+00(=)	2.339E+02	2.573E+02/9.878E+00
26	2.001E+02(-)	3.233E+02/6.917E+01(-)	2.000E+02(-)	3.179E+02/5.945E+01(-)	2.000E+02(-)	2.267E+02/4.614E+01(-)	2.000E+02(-)	2.543E+02/5.413E+01(-)	2.000E+02	2.263E+02/5.388E+01
27	6.686E+02(-)	9.676E+02/1.043E+02(-)	5.372E+02(-)	8.118E+02/1.672E+02(-)	3.031E+02(+)	4.158E+02/1.056E+02(+)	3.001E+02(+)	4.689E+02/1.305E+02(+)	4.040E+02	7.156E+02/1.371E+02
28	1.000E+02(+)	3.659E+02/2.844E+02(=)	1.000E+02(+)	3.897E+02/2.903E+02(-)	3.000E+02(=)	3.000E+02/0.000E+00(=)	3.000E+02(=)	3.218E+02/1.556E+02(=)	3.000E+02	3.422E+02/2.109E+02
-/=/+	24/0/4	25/3/0	20/2/6	22/5/1	14/4/10	13/4/11	18/3/7	15/9/4	-/-/-	-/-/-

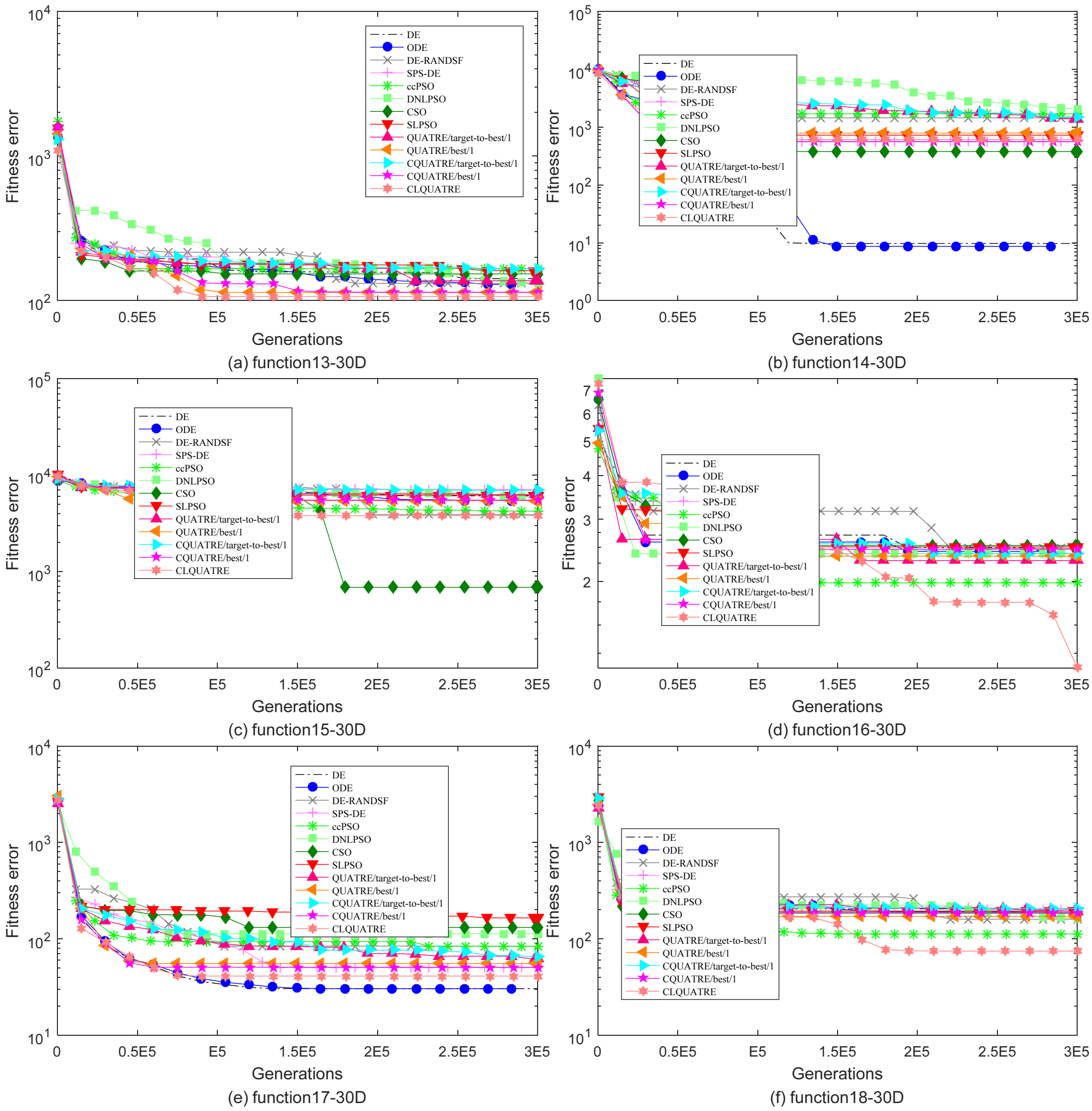

Figure S3. Comparison of the median of fitness errors for functions $f_{13} - f_{18}$ with 30D optimization

Table S4. Comparison results of CL-QUATRE with different σ values on CEC2013 test suite

30D	CL-QUATRE $\sigma = 0$		CL-QUATRE $\sigma = 0.1$		CL-QUATRE $\sigma = 0.2$		CL-QUATRE $\sigma = 0.3$		CL-QUATRE $\sigma = 0.4$	
	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std
1	0.000E+00(=)	0.000E+00/0.000E+00(=)	0.000E+00	0.000E+00/0.000E+00	0.000E+00(=)	1.338E-14/5.403E-14(=)	0.000E+00(=)	2.675E-14/7.399E-14(-)	0.000E+00(=)	8.025E-14/1.097E-13(-)
2	6.958E+04(+)	2.546E+05/1.279E+05(=)	7.130E+04	2.567E+05/1.419E+05	4.454E+04(+)	2.288E+05/1.103E+05(=)	5.643E+04(+)	2.126E+05/1.139E+05(=)	2.902E+04(+)	2.213E+05/1.007E+05(=)
3	6.086E-05(+)	7.024E+05/2.574E+06(=)	1.723E-01	1.152E+06/2.325E+06	8.683E-01(-)	3.030E+06/7.804E+06(=)	8.737E-01(-)	7.744E+06/2.615E+07(-)	5.648E+02(-)	6.243E+06/9.402E+06(-)
4	5.180E+00(-)	2.226E+01/1.198E+01(-)	2.884E+00	1.553E+01/1.063E+01	1.655E+00(+)	1.129E+01/7.071E+00(+)	1.239E+00(+)	8.740E+00/6.257E+00(+)	1.193E+00(+)	8.014E+00/6.098E+00(+)
5	0.000E+00(=)	1.115E-13/1.592E-14(=)	0.000E+00	1.092E-13/2.229E-14	0.000E+00(=)	1.115E-13/1.592E-14(=)	1.137E-13(-)	1.137E-13/0.000E+00(=)	1.137E-13(-)	1.271E-13/3.699E-14(-)
6	4.677E-07(-)	4.547E+00/8.148E+00(=)	4.850E-09	4.088E+00/7.516E+00	4.669E-07(-)	5.102E+00/7.993E+00(-)	8.596E-09(-)	1.227E+01/1.757E+01(-)	5.213E-07(-)	8.508E+00/9.544E+00(-)
7	1.817E+00(-)	1.072E+01/9.176E+00(=)	1.802E+00	1.583E+01/1.557E+01	2.079E+00(-)	1.906E+01/1.117E+01(-)	6.806E+00(-)	3.217E+01/1.708E+01(-)	1.259E+01(-)	3.416E+01/1.490E+01(-)
8	2.074E+01(+)	2.094E+01/6.123E-02(=)	2.075E+01	2.094E+01/5.935E-02	2.068E+01(+)	2.095E+01/6.536E-02(=)	2.065E+01(+)	2.093E+01/6.724E-02(=)	2.065E+01(+)	2.092E+01/8.051E-02(=)
9	5.906E+00(+)	2.098E+01/5.996E+00(=)	9.803E+00	1.955E+01/4.456E+00	8.206E+00(+)	1.889E+01/4.978E+00(=)	1.045E+01(-)	2.166E+01/4.519E+00(-)	1.461E+01(-)	2.228E+01/3.876E+00(-)
10	7.396E-03(-)	3.361E-02/2.312E-02(=)	0.000E+00	3.839E-02/2.471E-02	5.684E-14(-)	3.975E-02/2.358E-02(=)	7.396E-03(-)	4.674E-02/2.950E-02(=)	9.857E-03(-)	6.634E-02/5.406E-02(-)
11	5.970E+00(-)	1.434E+01/5.167E+00(=)	4.975E+00	1.317E+01/5.218E+00	5.970E+00(-)	1.574E+01/5.652E+00(-)	1.194E+01(-)	2.251E+01/6.411E+00(-)	1.094E+01(-)	3.209E+01/1.036E+01(-)
12	2.686E+01(-)	5.864E+01/1.756E+01(-)	2.288E+01	5.057E+01/1.294E+01	3.283E+01(-)	6.061E+01/2.045E+01(-)	3.283E+01(-)	6.469E+01/1.975E+01(-)	2.885E+01(-)	7.162E+01/2.024E+01(-)
13	5.459E+01(+)	1.064E+02/2.253E+01(=)	5.957E+01	1.085E+02/2.390E+01	5.581E+01(+)	1.055E+02/2.792E+01(=)	5.938E+01(+)	1.215E+02/2.725E+01(-)	5.522E+01(+)	1.201E+02/2.974E+01(-)
14	9.560E+01(+)	5.689E+02/2.140E+02(=)	1.652E+02	5.951E+02/2.214E+02	3.037E+02(-)	6.239E+02/2.056E+02(=)	1.674E+02(-)	5.686E+02/2.146E+02(=)	2.229E+02(-)	6.442E+02/2.375E+02(=)
15	2.580E+03(-)	3.665E+03/4.433E+02(=)	2.069E+03	3.718E+03/6.164E+02	2.627E+03(-)	3.719E+03/5.913E+02(=)	2.687E+03(-)	3.909E+03/5.408E+02(=)	2.410E+03(-)	3.875E+03/6.523E+02(=)
16	2.145E-01(+)	1.217E+00/5.426E-01(=)	2.879E-01	1.144E+00/5.438E-01	2.721E-01(+)	1.176E+00/5.119E-01(=)	2.395E-01(+)	9.589E-01/4.764E-01(=)	2.789E-01(+)	9.690E-01/5.427E-01(=)
17	1.300E+01(+)	4.106E+01/5.572E+00(=)	3.398E+01	4.122E+01/4.059E+00	2.365E+01(+)	4.305E+01/5.177E+00(-)	3.759E+01(-)	4.718E+01/6.568E+00(-)	3.744E+01(-)	5.174E+01/8.410E+00(-)
18	4.577E+01(-)	7.970E+01/1.558E+01(=)	4.374E+01	7.789E+01/1.604E+01	5.147E+01(-)	8.060E+01/1.523E+01(=)	5.375E+01(-)	8.543E+01/1.520E+01(-)	5.550E+01(-)	9.468E+01/2.264E+01(-)
19	9.153E-01(+)	2.136E+00/4.868E-01(=)	1.158E+00	2.331E+00/6.108E-01	1.315E+00(-)	2.540E+00/6.801E-01(=)	1.564E+00(-)	2.972E+00/8.889E-01(-)	1.752E+00(-)	3.467E+00/1.272E+00(-)
20	9.433E+00(-)	1.102E+01/7.629E-01(=)	9.020E+00	1.088E+01/6.988E-01	8.864E+00(+)	1.104E+01/6.582E-01(=)	9.435E+00(-)	1.118E+01/7.592E-01(=)	9.849E+00(-)	1.106E+01/7.024E-01(=)
21	2.000E+02(=)	3.009E+02/7.413E+01(=)	2.000E+02	3.055E+02/8.428E+01	2.000E+02(=)	2.974E+02/9.670E+01(=)	2.000E+02(=)	2.847E+02/6.943E+01(=)	2.000E+02(=)	3.096E+02/7.221E+01(=)
22	2.066E+02(+)	6.250E+02/2.219E+02(=)	2.085E+02	6.215E+02/2.451E+02	1.362E+02(+)	5.618E+02/2.253E+02(=)	2.549E+02(-)	7.663E+02/2.822E+02(-)	2.542E+02(-)	7.419E+02/2.490E+02(-)
23	2.406E+03(-)	4.028E+03/5.809E+02(=)	2.294E+03	4.024E+03/6.851E+02	1.621E+03(+)	4.064E+03/7.160E+02(=)	2.264E+03(+)	4.106E+03/6.635E+02(=)	2.077E+03(+)	4.074E+03/6.747E+02(=)
24	2.074E+02(-)	2.345E+02/1.253E+01(=)	2.042E+02	2.374E+02/1.302E+01	2.164E+02(-)	2.443E+02/1.314E+01(-)	2.238E+02(-)	2.486E+02/1.106E+01(-)	2.283E+02(-)	2.533E+02/1.236E+01(-)
25	2.382E+02(-)	2.588E+02/8.847E+00(=)	2.339E+02	2.573E+02/9.878E+00	2.469E+02(-)	2.662E+02/8.599E+00(-)	2.535E+02(-)	2.694E+02/1.064E+01(-)	2.554E+02(-)	2.737E+02/8.836E+00(-)
26	2.000E+02(-)	2.206E+02/4.832E+01(=)	2.000E+02	2.263E+02/5.388E+01	2.000E+02(+)	2.375E+02/6.191E+01(=)	2.000E+02(-)	2.231E+02/5.423E+01(=)	2.000E+02(+)	2.390E+02/6.746E+01(=)
27	4.403E+02(-)	6.767E+02/9.343E+01(=)	4.040E+02	7.156E+02/1.371E+02	5.530E+02(-)	7.718E+02/9.513E+01(-)	5.741E+02(-)	8.159E+02/1.058E+02(-)	5.924E+02(-)	8.389E+02/1.098E+02(-)
28	1.000E+02(+)	3.372E+02/2.081E+02(=)	3.000E+02	3.422E+02/2.109E+02	3.000E+02(=)	3.414E+02/2.071E+02(=)	1.000E+02(+)	3.834E+02/3.049E+02(=)	3.000E+02(=)	3.232E+02/1.654E+02(=)
-/=/+	14/3/11	2/26/0	-/-/-	-/-/-	13/4/11	8/19/1	19/2/7	15/12/1	18/3/7	17/10/1

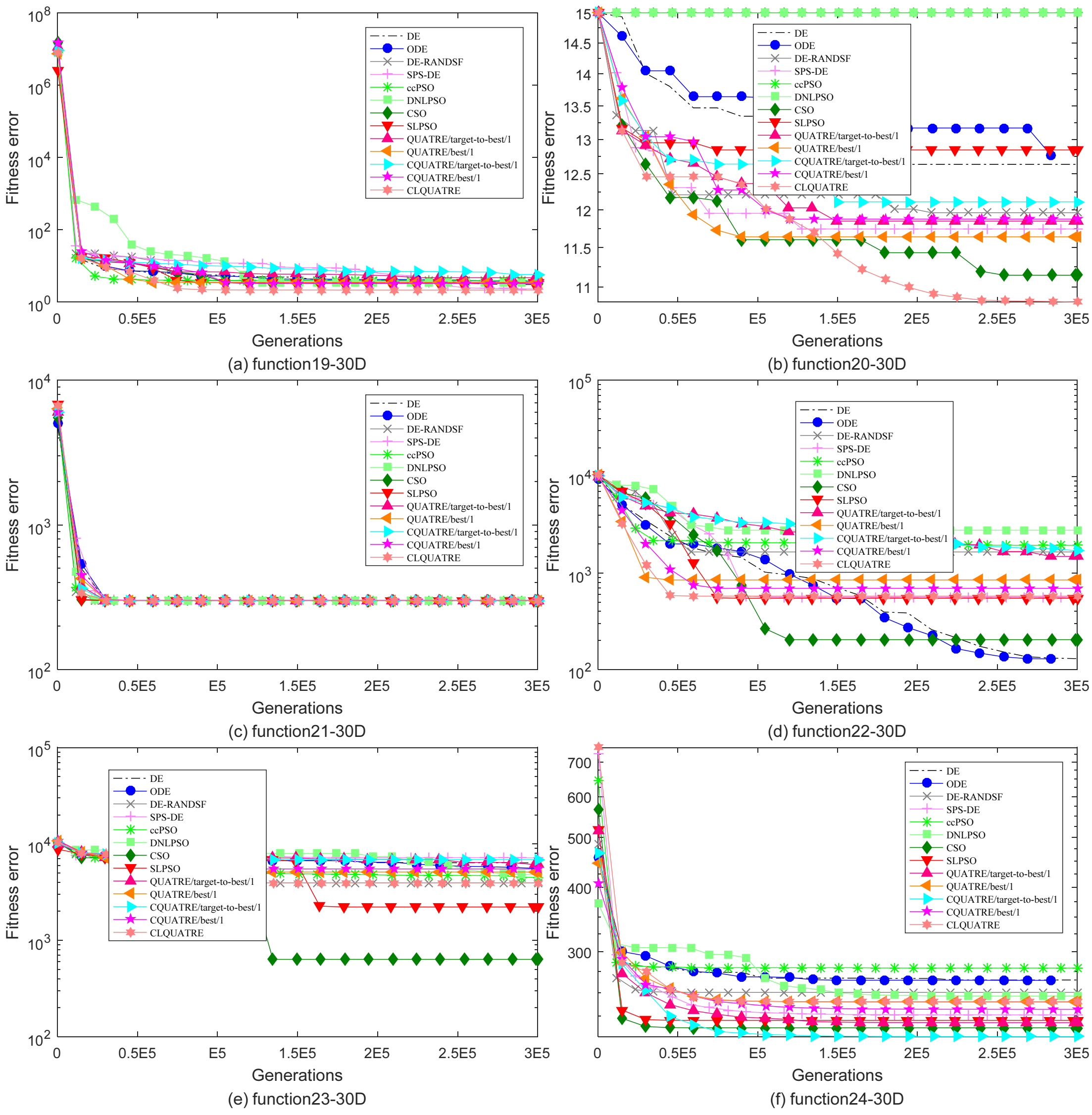

Figure S4. Comparison of the median of fitness errors for functions $f_{19} - f_{24}$ with 30D optimization

Table S5. Comparison results of CL-QUATRE with const F = 0.7 and stochastic scale factor on CEC2013 test suite

30D	CL-QUATRE with const F=0.7		CL-QUATRE with stochastic scale factor	
	Best	Mean/Std	Best	Mean/Std
1	0.0000E+00(=)	0.0000E+00/0.0000E+00(=)	0.0000E+00	0.0000E+00/0.0000E+00
2	1.3822E+05(-)	4.6815E+05/2.5626E+05(-)	7.1295E+04	2.5675E+05/1.4187E+05
3	1.1275E-08(+)	1.1854E+05/4.3248E+05(+)	1.7232E-01	1.1519E+06/2.3247E+06
4	2.7666E+01(-)	1.3883E+02/6.2687E+01(-)	2.8835E+00	1.5529E+01/1.0634E+01
5	0.0000E+00(=)	1.0254E-13/3.4143E-14(=)	0.0000E+00	1.0923E-13/2.2287E-14
6	1.5631E-04(-)	5.7865E+00/8.3725E+00(-)	4.8504E-09	4.0875E+00/7.5156E+00
7	7.2509E-01(+)	8.2124E+00/8.5653E+00(+)	1.8021E+00	1.5827E+01/1.5573E+01
8	2.0827E+01(-)	2.0963E+01/4.2731E-02(-)	2.0745E+01	2.0938E+01/5.9352E-02
9	9.1760E+00(+)	2.1089E+01/6.6849E+00(=)	9.8027E+00	1.9550E+01/4.4562E+00
10	0.0000E+00(=)	1.9124E-02/1.3618E-02(+)	0.0000E+00	3.8387E-02/2.4707E-02
11	6.9647E+00(-)	1.8436E+01/5.4254E+00(-)	4.9748E+00	1.3169E+01/5.2179E+00
12	3.0185E+01(-)	5.7688E+01/1.7210E+01(=)	2.2884E+01	5.0567E+01/1.2942E+01
13	6.2704E+01(-)	1.0792E+02/2.3437E+01(=)	5.9568E+01	1.0851E+02/2.3899E+01
14	1.6522E+02(+)	6.7802E+02/2.6816E+02(=)	1.6524E+02	5.9506E+02/2.2140E+02
15	2.7448E+03(-)	4.1960E+03/7.3712E+02(-)	2.0689E+03	3.7180E+03/6.1642E+02
16	8.4836E-01(-)	1.6972E+00/4.4796E-01(-)	2.8790E-01	1.1444E+00/5.4382E-01
17	1.9536E+01(+)	4.7153E+01/6.0525E+00(-)	3.3983E+01	4.1216E+01/4.0591E+00
18	6.7457E+01(-)	1.0366E+02/1.9745E+01(-)	4.3739E+01	7.7889E+01/1.6044E+01
19	1.3785E+00(-)	2.4359E+00/5.6098E-01(=)	1.1580E+00	2.3311E+00/6.1075E-01
20	1.0142E+01(-)	1.1354E+01/6.6262E-01(-)	9.0198E+00	1.0878E+01/6.9883E-01
21	2.0000E+02(=)	3.0009E+02/6.9891E+01(=)	2.0000E+02	3.0547E+02/8.4277E+01
22	2.1970E+02(-)	6.7634E+02/2.0488E+02(=)	2.0850E+02	6.2152E+02/2.4512E+02
23	2.5123E+03(-)	4.1336E+03/8.0197E+02(=)	2.2942E+03	4.0236E+03/6.8506E+02
24	2.0080E+02(+)	2.3390E+02/1.4465E+01(=)	2.0417E+02	2.3741E+02/1.3025E+01
25	2.3855E+02(-)	2.5378E+02/7.5879E+00(=)	2.3390E+02	2.5729E+02/9.8778E+00
26	2.0001E+02(-)	2.3952E+02/6.2095E+01(-)	2.0001E+02	2.2629E+02/5.3875E+01
27	3.2613E+02(+)	6.8327E+02/1.4038E+02(=)	4.0398E+02	7.1562E+02/1.3706E+02
28	1.0000E+02(+)	3.5686E+02/2.4814E+02(=)	3.0000E+02	3.4219E+02/2.1088E+02
-/=/+	16/4/8	11/14/3	-/-/-	-/-/-

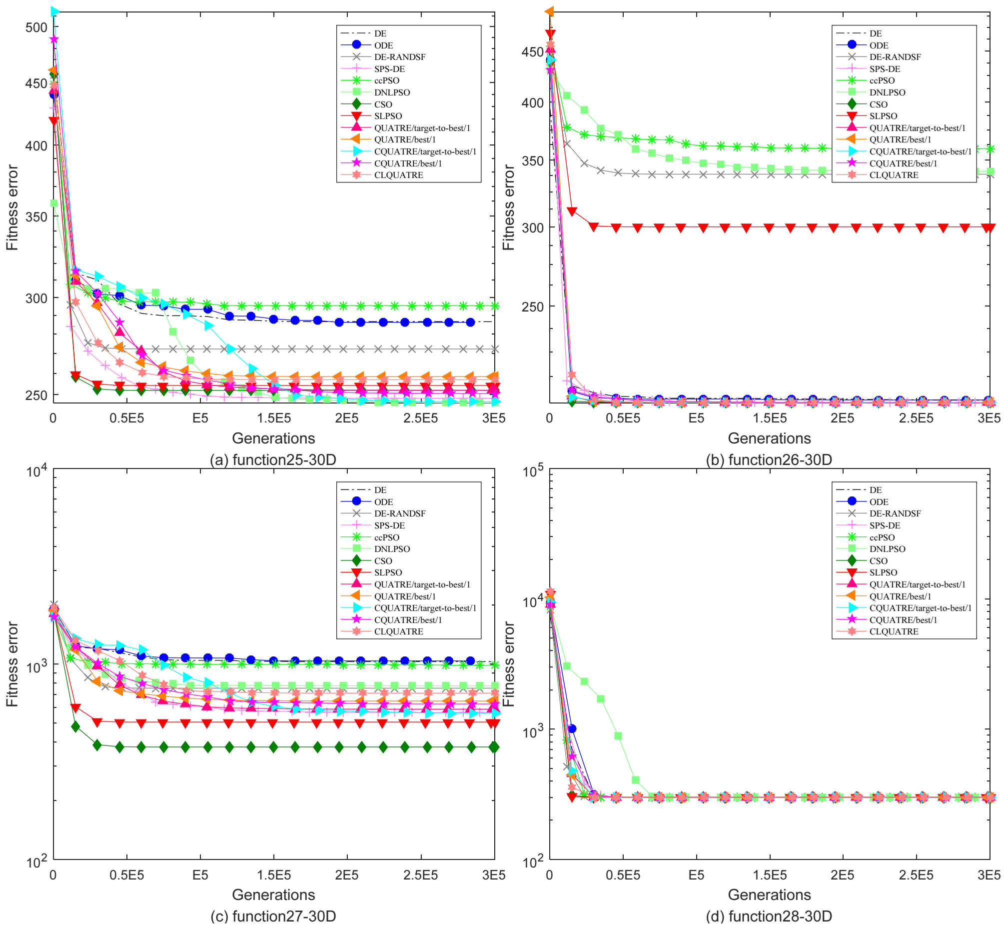

Figure S5. Comparison of the median of fitness errors for functions $f_{25} - f_{28}$ with 30D optimization

Table S6. Comparison results of CL-QUATRE with fixed F = 0.7 and different mutation strategies on CEC2013 test suite

30D	Best	Mean/Std	Best	Mean/Std
1	0.0000E+00(=)	0.0000E+00/0.0000E+00(=)	0.0000E+00	0.0000E+00/0.0000E+00
2	1.0329E+05(+)	3.5700E+05/1.6447E+05(+)	1.3822E+05	4.6815E+05/2.5626E+05
3	1.4080E-03(-)	4.5240E+05/1.1803E+06(-)	1.1275E-08	1.1854E+05/4.3248E+05
4	6.2328E+00(+)	3.9319E+01/2.2387E+01(+)	2.7666E+01	1.3883E+02/6.2687E+01
5	0.0000E+00(=)	8.2479E-14/5.1240E-14(+)	0.0000E+00	1.0254E-13/3.4143E-14
6	2.7300E-06(+)	5.6334E+00/8.4309E+00(=)	1.5631E-04	5.7865E+00/8.3725E+00
7	3.6896E-01(+)	9.9847E+00/9.9559E+00(=)	7.2509E-01	8.2124E+00/8.5653E+00
8	2.0863E+01(-)	2.0981E+01/4.2993E-02(-)	2.0827E+01	2.0963E+01/4.2731E-02
9	6.4787E+00(+)	2.0603E+01/7.0574E+00(=)	9.1760E+00	2.1089E+01/6.6849E+00
10	7.3960E-03(-)	2.9227E-02/1.4111E-02(-)	0.0000E+00	1.9124E-02/1.3618E-02
11	9.9496E+00(-)	1.9729E+01/4.8372E+00(=)	6.9647E+00	1.8436E+01/5.4254E+00
12	2.5869E+01(+)	6.1069E+01/1.5119E+01(=)	3.0185E+01	5.7688E+01/1.7210E+01
13	4.0042E+01(+)	1.0609E+02/3.0523E+01(=)	6.2704E+01	1.0792E+02/2.3437E+01
14	2.4898E+02(-)	7.2861E+02/2.7106E+02(=)	1.6522E+02	6.7802E+02/2.6816E+02
15	3.2879E+03(-)	4.6204E+03/6.2821E+02(-)	2.7448E+03	4.1960E+03/7.3712E+02
16	1.0541E+00(-)	1.9358E+00/4.4794E-01(-)	8.4836E-01	1.6972E+00/4.4796E-01
17	4.0140E+01(-)	4.8992E+01/6.3183E+00(=)	1.9536E+01	4.7153E+01/6.0525E+00
18	7.7368E+01(-)	1.2566E+02/2.4731E+01(-)	6.7457E+01	1.0366E+02/1.9745E+01
19	1.5714E+00(-)	2.8544E+00/7.5371E-01(-)	1.3785E+00	2.4359E+00/5.6098E-01
20	9.7825E+00(+)	1.1157E+01/6.5072E-01(=)	1.0142E+01	1.1354E+01/6.6262E-01
21	2.0000E+02(=)	2.9140E+02/7.0690E+01(=)	2.0000E+02	3.0009E+02/6.9891E+01
22	2.7999E+02(-)	8.0529E+02/2.4893E+02(-)	2.1970E+02	6.7634E+02/2.0488E+02
23	2.9481E+03(-)	4.4669E+03/8.1333E+02(=)	2.5123E+03	4.1336E+03/8.0197E+02
24	2.0129E+02(-)	2.2823E+02/1.4438E+01(=)	2.0080E+02	2.3390E+02/1.4465E+01
25	2.4306E+02(-)	2.5583E+02/7.0269E+00(=)	2.3855E+02	2.5378E+02/7.5879E+00
26	2.0001E+02(-)	2.4392E+02/6.3127E+01(=)	2.0001E+02	2.3952E+02/6.2095E+01
27	4.9750E+02(-)	6.7068E+02/9.2778E+01(=)	3.2613E+02	6.8327E+02/1.4038E+02
28	3.0000E+02(-)	3.6096E+02/2.4656E+02(=)	1.0000E+02	3.5686E+02/2.4814E+02
-/=/+	17/3/8	17/8/3	-/-/-	-/-/-

Table S7. Comparison results of the four competitive algorithm on CEC2014 test suite

30D	CSO		C-QUATRE/best/1		C-QUATRE/target-to-best/1		CL-QUATRE	
	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std	Best	Mean/Std
1	8.1407E+04(-)	3.8432E+05/2.2202E+05(-)	1.7947E+04(-)	2.7360E+05/2.2573E+05(-)	1.7151E+04(-)	1.5464E+05/1.3702E+05(-)	5.5387E+03	1.0026E+05/9.3199E+04
2	5.4019E+00(-)	1.0893E+04/9.9208E+03(-)	0.0000E+00(=)	2.7864E-15/8.5358E-15(=)	0.0000E+00(=)	0.0000E+00/0.0000E+00(+)	0.0000E+00	2.2292E-15/7.7172E-15
3	6.5667E+02(-)	7.8373E+03/6.1839E+03(-)	0.0000E+00(=)	4.4583E-15/1.5434E-14(=)	0.0000E+00(=)	1.1146E-15/7.9597E-15(+)	0.0000E+00	1.1146E-14/2.2793E-14
4	4.5880E-02(-)	4.6253E+01/3.2678E+01(-)	6.3852E-06(-)	4.9804E+00/1.7213E+01(-)	2.5464E-08(-)	1.2586E+00/8.8762E+00(-)	2.1407E-10	6.2465E+00/1.9136E+01
5	2.0837E+01(-)	2.0950E+01/4.7154E-02(-)	2.0339E+01(-)	2.0499E+01/8.0211E-02(-)	2.0539E+01(-)	2.0631E+01/3.8242E-02(-)	2.0000E+01	2.0148E+01/1.4199E-01
6	0.0000E+00(+)	3.9243E-01/6.5183E-01(+)	0.0000E+00(+)	2.0057E+00/1.5626E+00(+)	0.0000E+00(+)	1.1650E+00/2.9510E+00(+)	6.1390E-01	3.8072E+00/2.2332E+00
7	0.0000E+00(=)	0.0000E+00/0.0000E+00(+)	0.0000E+00(=)	3.8635E-03/6.8421E-03(+)	0.0000E+00(=)	1.8838E-03/4.3665E-03(+)	0.0000E+00	7.2876E-03/1.0507E-02
8	2.9849E+00(+)	9.0522E+00/2.7157E+00(+)	7.9597E+00(-)	1.6992E+01/4.1702E+00(-)	1.8427E+01(-)	2.5814E+01/3.3795E+00(-)	4.9748E+00	1.2018E+01/4.2252E+00
9	2.9849E+00(+)	9.6982E+00/3.2298E+00(+)	3.4235E+01(-)	6.6201E+01/1.9571E+01(-)	1.1017E+02(-)	1.2250E+02/6.5771E+00(-)	2.6864E+01	5.0567E+01/1.6714E+01
10	1.1893E+01(+)	1.1397E+02/1.2864E+02(+)	1.0857E+01(+)	3.3686E+02/2.1361E+02(=)	4.1943E+02(-)	7.5250E+02/1.6502E+02(-)	2.0223E+01	2.6247E+02/1.5381E+02
11	1.3579E+01(+)	3.2291E+02/2.1590E+02(+)	1.7068E+03(-)	3.2658E+03/7.9766E+02(-)	4.6078E+03(-)	5.0075E+03/1.8029E+02(-)	1.2311E+03	2.3900E+03/5.0488E+02
12	1.5812E+00(-)	2.3842E+00/2.8879E-01(-)	9.6575E-02(-)	4.7391E-01/2.3655E-01(-)	7.8824E-01(-)	1.1759E+00/1.1832E-01(-)	2.8913E-02	1.7801E-01/9.7759E-02
13	8.3716E-02(+)	1.2844E-01/2.0969E-02(+)	1.1521E-01(+)	3.2210E-01/8.3624E-02(-)	2.1671E-01(-)	3.0363E-01/4.2262E-02(-)	1.1917E-01	2.1068E-01/5.6801E-02
14	3.1091E-01(-)	4.0183E-01/4.1025E-02(-)	1.3951E-01(-)	2.7478E-01/8.3442E-02(-)	1.5802E-01(-)	2.3777E-01/3.2262E-02(=)	1.2346E-01	2.2209E-01/4.2415E-02
15	2.2016E+00(+)	3.1767E+00/4.4504E-01(+)	4.4891E+00(-)	8.0362E+00/2.0006E+00(-)	9.1330E+00(-)	1.1679E+01/8.1214E-01(-)	2.2161E+00	3.7543E+00/1.0427E+00
16	9.7696E+00(-)	1.0942E+01/3.7368E-01(-)	7.8305E+00(+)	1.0193E+01/9.0060E-01(=)	1.0559E+01(-)	1.1573E+01/2.9155E-01(-)	8.7490E+00	1.0040E+01/5.8179E-01
17	9.0931E+03(-)	1.2401E+05/7.1995E+04(-)	7.7572E+02(-)	3.1834E+03/2.5085E+03(=)	1.7224E+02(+)	1.6796E+03/1.5302E+03(+)	7.5669E+02	2.1934E+03/1.4686E+03
18	1.2273E+01(+)	1.2868E+03/1.8063E+03(-)	1.4675E+01(-)	3.9201E+01/1.4255E+01(+)	2.5683E+01(-)	6.1105E+01/1.5241E+01(=)	1.4644E+01	5.7982E+01/3.5600E+01
19	3.2243E+00(-)	5.4685E+00/8.6159E-01(-)	2.9431E+00(-)	4.8061E+00/1.5491E+00(=)	3.6698E+00(-)	4.9599E+00/1.0869E+00(=)	2.4224E+00	4.6878E+00/1.3893E+00
20	2.4072E+03(-)	1.4240E+04/7.1595E+03(-)	6.9897E+00(-)	2.5244E+01/1.4094E+01(=)	2.4765E+01(-)	3.5410E+01/5.3531E+00(-)	5.9605E+00	2.2250E+01/1.0606E+01
21	8.5820E+03(-)	9.8568E+04/6.3024E+04(-)	1.4044E+01(+)	6.7406E+02/6.3685E+02(-)	7.3646E+01(+)	6.4304E+02/2.9087E+02(-)	1.0339E+02	4.2243E+02/1.9433E+02
22	2.0547E+01(+)	1.1741E+02/6.7068E+01(+)	2.2245E+01(-)	3.2429E+02/1.2192E+02(-)	2.4243E+01(-)	1.9506E+02/1.2111E+02(=)	2.2010E+01	2.0304E+02/1.1505E+02
23	3.1524E+02(=)	3.1524E+02/6.0023E-12(=)	3.1524E+02(=)	3.1524E+02/4.0186E-13(=)	3.1524E+02(=)	3.1524E+02/4.0186E-13(=)	3.1524E+02	3.1524E+02/4.0186E-13
24	2.2289E+02(-)	2.2664E+02/3.4150E+00(=)	2.2172E+02(+)	2.2550E+02/3.7114E+00(+)	2.2126E+02(+)	2.2526E+02/4.0551E+00(+)	2.2177E+02	2.2886E+02/6.2089E+00
25	2.0337E+02(-)	2.0526E+02/1.1425E+00(-)	2.0258E+02(+)	2.0335E+02/7.2900E-01(+)	2.0260E+02(+)	2.0321E+02/4.9814E-01(+)	2.0272E+02	2.0387E+02/9.4971E-01
26	1.0009E+02(+)	1.1188E+02/3.2515E+01(+)	1.0016E+02(-)	1.0227E+02/1.3967E+01(-)	1.0023E+02(-)	1.0031E+02/4.3959E-02(-)	1.0011E+02	1.0021E+02/5.1739E-02
27	3.0000E+02(+)	3.5430E+02/5.2027E+01(+)	3.0000E+02(+)	3.8649E+02/4.6338E+01(=)	3.0000E+02(+)	3.6671E+02/4.6486E+01(+)	3.0051E+02	4.0901E+02/7.2306E+01
28	7.7467E+02(-)	8.5825E+02/3.9515E+01(+)	6.7009E+02(+)	8.4275E+02/6.5492E+01(+)	6.6630E+02(+)	8.7982E+02/1.3763E+02(=)	7.5624E+02	8.9768E+02/8.3633E+01
29	1.0967E+03(-)	1.6449E+03/4.5744E+02(-)	3.9169E+02(-)	2.1557E+05/1.5337E+06(=)	1.9620E+02(+)	7.4462E+02/1.6602E+02(+)	3.2226E+02	3.6882E+05/1.8412E+06
30	1.5302E+03(-)	2.9097E+03/9.7720E+02(-)	4.3714E+02(+)	1.4285E+03/7.0791E+02(=)	4.5565E+02(+)	1.0991E+03/4.1289E+02(+)	5.5235E+02	1.6331E+03/7.5079E+02
-/=/+	17/2/11	16/2/12	16/4/10	13/11/6	17/4/9	14/6/10	-/-/-	-/-/-