Center-Based Initialization of Cooperative Co-evolutionary Algorithm for Large-scale Optimization

Sedigheh Mahdavi
Department of Electrical, Computer, and Software Engineering
University of Ontario Institute of Technology (UOIT)
Oshawa, Canada
Email: Sedigheh.Mahdavi@uoit.ca

Shahryar Rahnamayan
Department of Electrical, Computer, and Software Engineering
University of Ontario Institute of Technology (UOIT)
Oshawa, Canada
Email: Shahryar.Rahnamayan@uoit.ca

Kalyanmoy Deb
Department of Electrical and Computer Engineering
Michigan State University
East Lansing, USA
Email: kdeb@egr.msu.edu

COIN Report Number 2015024

Abstract—**Cooperative Coevolution (CC) framework has become a powerful approach to solve large-scale global optimization problems effectively. Although a number of significant modifications of CC algorithms have been introduced in recent years, the theoretical studies of population initialization strategies in the CC framework are quite limited so far. The population initialization strategies can help a population-based algorithm to start with better candidate solutions for achieving better results. In this paper, we propose a CC algorithm with population initialization strategies based on the center region to improve its performance. Three population initialization strategies, namely, center-based normal distribution sampling, central golden region, and hybrid random-center normal distribution sampling are utilized in the CC framework. These population initialization strategies attempt to generate points around center-point with different schemes. The performance of the proposed algorithm is evaluated on CEC-2013 LSGO benchmark functions. Simulation results confirm that the proposed algorithm obtains a promising performance on the majority of the nonseparable high dimension benchmark functions.**

I. Introduction

Large-Scale Global Optimization (LSGO) known as challenging and demanding research field which has a variety of applications for solving real-world problems with a large number of decision variables. Most of the LSGO studies focus on two main branches of metaheuristic algorithms: Cooperative Co-evolution (CC) algorithms [1], [2], and non-decomposition based methods. The non-decomposition based methods increase the performance of standard metaheuristic algorithms to tackle LSGO problems by focusing on especial alterations such as defining new mutation, selection, and crossover operators, designing and using local search, opposition-based learning, sampling operator, hybridization, or variable population size methods [3]. Cooperative Coevolution (CC) algorithms [1], [2] have been designed and applied to solve the problems with a large number of decision variables. CC algorithms used the divide-and-conquer strategy to divide a high-dimensional decision vector into some low-dimensional subcomponents. The efficiency of the CC algorithms depends especially on the utilized grouping method to decompose a large scale problem into several low-dimensional subcomponents. A major challenge of CC algorithms is handling nonseparable problems and developing an optimal grouping method. The optimal grouping method allocates interacting variables to the same subcomponent such that the interactions among different subcomponents are minimal. In recent years, several CC algorithms have been proposed to solve nonseparable LSGO problems.

Another valuable research direction for solving LSGO problems is the population initialization methods efficiently for utilizing opposition-based computation concept, such as opposition-based differential evolution (ODE) [4] which was proposed to enhance differential evolution algorithm. The main concept of OBL is considering both the current candidate solution and its corresponding opposite current candidate solution simultaneously. Recently, the center-point sampling concept was proposed by Rahnamayan and Wang (2009) [5]. They have shown that the probability of closeness to an unknown solution for the center point is significantly higher than other points. The valuable property of the center point gives for the generated points closer to the center point the higher chance to be closer to an unknown solution. In some recent research works [6], [7], the center-point sampling method was utilized in the metaheuristic algorithms to accelerate their convergence speed. All research works on the population initialization strategy have shown that the enhanced versions of initialization are promising in improving of population-based algorithms' performance. The theoretical studies are still quite infant on considering the population initialization strategies to solve LSGO problems. Therefore, studying the various strategies to generate initial solutions in the LSGO problems would be beneficial.

In this paper, a CC framework with the population initialization strategies based on the property of the center point is proposed. The population initialization strategies based on the center point, namely, center-based normal distribution sampling, central golden region, and hybrid random-center normal distribution sampling, are utilized in to the CC framework.

The center-based normal distribution sampling strategy uses a normal distribution to generate points around the center point. In the central golden region strategy, the range of search space is limited to a special region for sampling points close to the center point. The performance of the proposed CC methods are evaluated on the shifted and rotated CEC-2013 LSGO benchmark functions for D=1000 [8]. Since the nonseparable benchmark functions in the CEC-2013 LSGO benchmark functions are shifted and rotated, it is not supposed to be a bias towards the center of search space. The results confirmed that the proposed population initialization strategies accelerate CC methods to solve nonseparable LSGO problems.

The organization of the rest of the paper is as follows. Section II presents a background review. Section III describes the details of the proposed method. Section IV presents the experimental results and discussion. Finally, the paper is concluded in Section V.

II. BACKGROUND REVIEW

A. *Cooperative Co-evolution*

Potter and Jong [1], [2] proposed the first CC framework in 1994. The classical steps of the general CC framework can be summarized as follows:

- Decompose a high-dimensional variable vector into some low-dimensional subcomponents.
- Evolve each subcomponent by an optimization algorithm for a predefined number of generations using a round-robin strategy.
- Merge the solutions of all subcomponents to construct n-dimensional candidate solutions to evaluate the individuals in each of the subcomponents.
- Stop the evolutionary process if termination condition is satisfied.

Liu et al. [9] applied the CC framework with FEP (Fast Evolutionary Programming), called FEPCC, to solve optimization problems with up to 1000 dimensions. Also, they confirmed that the performance of the classical CC algorithms deteriorates rapidly on nonseparable problems. A challenging part of the CC algorithms is developing a proper grouping methods such that it can identify the interactions among variables in the nonseparable problems. When interacting variables are placed in different subcomponents, the CC algorithms perform poorly. The reason for this behavior of CC algorithms is that the influence of a variable on the fitness value in one subcomponent depends on other variables in different subcomponents; and subcomponents cannot be evolved independently.

Over the past decades, several different grouping strategies have been proposed to improve the performance of CC algorithms [3]. In [11], a random decomposition method (DECC-G) was proposed which decomposes randomly an n-dimensional problem into multiple low-dimensional subcomponents and utilizes an adaptive weighting method to tune further candidate solutions. A Multilevel CC (MLCC) algorithm was proposed in [10] to find an appropriate group size for subcomponents according to the performance history of the decomposer through the evolution process. Ray and Yao introduced the correlation matrix based methods [26], [12] which allocate variables to the separate subcomponents based on discovered correlations among the variables. A CC-based method with Variable Interaction Learning (CCVIL) was proposed in [13] to find interactions among variables. The Dependency Identification (DI) techniques [17], [18] were introduced to construct subcomponents according to the design of an internal minimization problem derived from the concept of partially separable function.

In [25], a statistical model was introduced based on the estimated probability value of the variable influence. For fully separable problems, in [20] a modification of the MLCC algorithm (MLSoft) was introduced with using reinforcement learning method to adaptively find the appropriate size value of subcomponents. In [19], a decomposition method based on high dimensional model representation method (DM-HDMR) was proposed by using the first-order RBF-HDMR proposed by Shan and Wang [23]. For considering the imbalance of subcomponents, two Contribution Based Cooperative Co-evolution (CBCC) methods, CBCC1 and CBCC2, were proposed in [24]. Recently, in [16], the differential grouping approach (DECC-DG) was introduced which is mathematically derived from the description of the partial separability. They defined and proved a theorem to detect the pairwise interaction of nonseparable variables. Based on this theorem, if the change in the objective function with respect to one variable depends on another variable; then two variables are nonseparable. We select the differential grouping (DG) and ideal methods as grouping method in the CC framework. In the ideal grouping method, subcomponents are manually constructed using the knowledge of benchmark functions [16].

B. *CENTER-BASED SAMPLING*

Rahnamayan and Wang introduced the concept of center-based sampling in [5]. Given the interval of search space as $[a, b]$, the center of the interval is formulated for dimensions 1 and n as follows:
For 1-dimension:

$$c = (a + b)/2$$

For n-dimension:

$$c_i = (a_i + b_i)/2,$$

where $i = 1 \dots D$ and D is the dimension of the problem. For a black-box problem, they investigated the probability of closeness to an unknown solution for the points of the search space with the Euclidean distance as the closeness measure. Monte-Carlo simulation was performed to compute this probability. They observed that the probability of closeness to an unknown solution for the center point is higher than other points and for the higher dimensions, this probability value increases extremely and approaches to one.

In the population-based algorithms, the center point cannot be utilized as the initialization population because it is a single unique point. On the other hand, a center-based region in the middle part of the interval was introduced which generated samples in the center-based region can be utilized for population-based algorithms [6], [5]. According to Fig. 1 [5], it can be seen that for the higher dimensions, the probability of closeness to an unknown optimum solution grows drastically in the center-based region. They implement a series of Monte-Carlo simulations to calculate the probability of closeness of the point x to an unknown optimum solution, compared to a second random point. Fig. 1 indicates the results of the simulations.

The Monte-Carlo simulations and Fig. 1 are described briefly as follow. On the space diagonal, the x values are uniformly disturbed from 0 to 1 with the stepsize 10^{-3} and then for all points the probability of closeness to unknown optimum solution is computed. For dimension n, the x values on the space diagonal are as follows:

$$(0,0,\ldots,0),(0+10^{-3},\ldots,0+10^{-3}),$$
$$(2\times 10^{-3},\ldots,2\times 10^{-3}),\ldots,(1,1,1,\ldots,1)$$

For each point x, 10^6 trials are considered, so an unknown optimum solution (s) and a second random point are generated uniform randomly in the search space. In each trial, the distance of the point x and second random point from s are computed. The closeness probability of point x (p_x) is computed as follows: the number of trials (N_t) which x is closer to an unknown optimum solution than the second random point divided by the number of total trials. Two probabilities are computed as follows:

$$p_x = p[d(x,s) < d(r,s)] = N_t/10^6,$$
$$p_r = p[d(r,s) < d(x,s)] = (10^6 - N_t)/10^6,$$

where p_r is the closeness probability of the second random point form an unknown optimum solution. Based on this definition, it is obvious that summation of two probability values are equal to 1 ($p_x + p_r = 1$). As it can seen from Fig. 1, the uniformly generated samples in the center-based region have the highest probability of closeness to an unknown optimum solution, especially in the interval $[0.2, 0.8]$ in an interval $[0, 1]$, i.e., 60% of the search space. In this simulations, the points are on the space diagonal, i.e., an N-dimensional vector with the same value for all elements. It can be seen from Fig. 1, the curves turned sharply to the almost a flat region (step-wise) by increasing dimension in the interval $[0.2, 0.8]$.

Now, we want to investigate the probability of the closeness of the center point to the optimum solution, compared to a second random point when the optimum solution is in the corner (the worst case scenario). Fig. 2 indicates the results of Monte-Carlo simulations for the worst case scenario. As it can seen from Fig. 2, the probability of the closeness of the center point to the optimum solution is increased when dimension is increased, especially dimensions higher than dimension 200. It means even for the worse case scenario, the center-based point has almost 100% chance to be closer to the optimum solution for higher dimensions.

III. Center-Point-Based Cooperative Co-evolutionary

In this section, three population initialization strategies based on the center point are utilized in the CC framework. This paper attempts to use the center region at the initialization step to improve the performance of the CC algorithms for solving nonseparable LSGO problems. In the CC framework, the uniform random initialization is the most commonly used

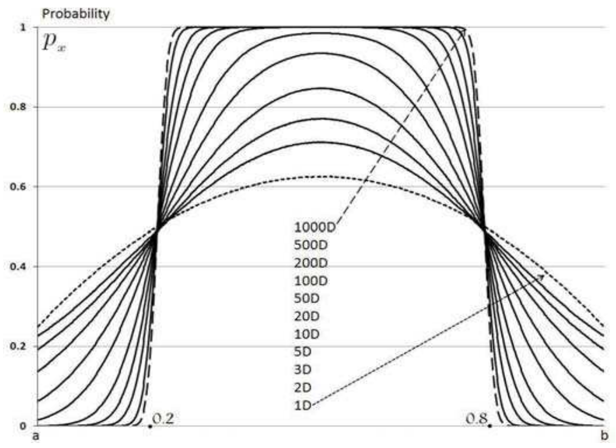


Fig. 1: Probability of closeness of candidate-solution to an unknown solution for different points in the interval [5]. Utilizing a vector with the same value for all its dimensions on the space diagonal is a proper way to provide $2D$ plots for the D dimensional search space.

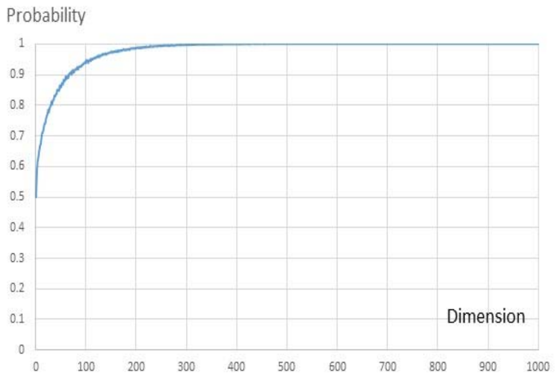


Fig. 2: Probability of center-point closeness to the optimum solution in the corner, worst case scenario (compared to a uniformly generated random point) versus dimension of search space.

strategy to create an initialization population. Since the general assumption in the CC methods is that there is no available information about the problem (i.e., black-box problem), the center-based sampling strategy by using only the central region can make more promising in solving nonseparable LSGO problems efficiently.

In following, three population initialization strategies, namely, center-based normal distribution sampling (CNS), central golden region (CGR), and hybrid random-center normal distribution sampling (HRCN), are described in details. The CGR strategy generates uniform-random points in the center-based region which are located in the middle 60% interval of the search space, proposed in [5]. This strategy samples points in a specific range of the entire space while it may not flexible enough for all kinds of problems. In CNS strategy, points in the initialization population are sampled with a normal distribution which its mean is the center-point. The generated points in the CNS strategy tends to be around the center point with the symmetric way due to using normal distribution. In the HRCN, the half of population is created with the normal distribution sampling strategy and another half is randomly sampled using the uniform distribution.

Fig. 3 indicates three population initialization strategies based on the center point and the random population initialization strategy to generate 100 points in the $2D$ interval of search space as $[-1,1]^2$. The dark black point represents the center of the search space. Fig. 3 (a) shows the random initialization population strategy which points are uniformly randomly generated in the all range of the search space. As seen in the Fig. 3 (b), in the CGR strategy, the probability of falling random point around the center point increases with limiting the interval of search space to the middle 60% interval of the search space. While Fig. 3 (c) shows how the CNS strategy can generate more points around the center point and also cover more range of search space than the CGR strategy. As seen in the Fig. 3 (d), in the HRCN sampling strategy, the generated points in the neighborhood of the center point are less than the CNS strategy but this strategy try to cover more range of the search space.

In [21], it was mentioned that for generating the values of a normal distribution in a range between 0 and 1, the points should sampled from a normal distribution with the mean value 0.5 and the standard deviation $1/6$. With this normal distribution, the probability of falling a normally distributed random point within the interval $[0,1]$ is 99.73%. Therefore, the CNS strategy generates points with a normal distribution with following mean μ and the standard deviation σ:
For 1-dimension:

$$\mu = (a+b)/2\ , \quad \sigma = (b-a)/6$$

For n-dimension:

$$\mu_i = (a_i+b_i)/2\ , \quad \sigma_i = (b_i-a_i)/6$$

In this strategy, the probability of falling a normally distributed random point within the interval $[a,b]$ is 99.73%. When points are located out of valid range, the new points are generated to replace with them. In this paper, all steps of CC framework are similar to a traditional CC framework but the the population initialization step uses the mentioned population initialization strategies instead of the random initialization strategy. The CC framework used two grouping methods, namely, the differential and ideal grouping methods. A self-adaptive differential evolution with the neighborhood search algorithm (namely SaNSDE) [22] is used as the subcomponent optimization algorithm in the CC framework.

IV. EXPERIMENTAL RESULTS

A. Setup of experiments

To analyze the performance of three population initialization strategies based on the center point, we compare three CC algorithms with CNS, CGR, and HRCN strategies, namely, CC-CNS, CC-CGR, and CC-HRCN, respectively, against the CC algorithms with the random initialization strategy (CC-R) on the ten nonseparable LSGO problems (f_4-f_{11}, and f_{13}-f_{14}) of the CEC-2013 LSGO benchmark functions with D=1000. Experiments were conducted with two grouping methods, DG and ideal grouping for CC algorithms. Also, three population size (NP), i.e., 50, 100, and 200 are considered to analyze the behaviors of population initialization strategies. A two-tailed t-test at a 0.05 significance level is performed among the CC algorithms with the different population initialization strategies and CC-R algorithm. The t-test is used to determine whether the mean of two populations are statistically different. The t-test reports a p-value, if this p-value become greater than significance level value, there is no significant difference between the mean of the two populations; otherwise there is a significant difference. Symbols '†', '‡', and '$\approx$' denote the compared algorithms are worse than, better than, or similar to the CC-R algorithm, respectively and the best obtained values are highlighted in bold. In this study, the maximum number of evaluations was set to 3×10^6, and all algorithms were evaluated for 25 independent runs and the results were recorded.

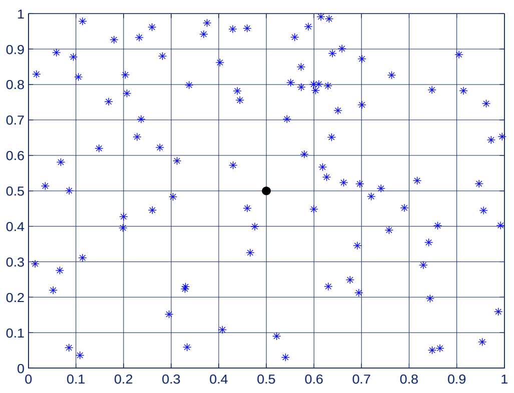
B. Numerical Results

The mean and the standard deviation of the obtained error values by CC-CNS, CC-CGR, CC-HRCN, and CC-R algorithms are summarized in Tables I and III. Also, we compute a rate value $rate_{imp}$, to reflect the relative improvement obtained for each population initialization strategy, X, based on the center point for each function as follows:

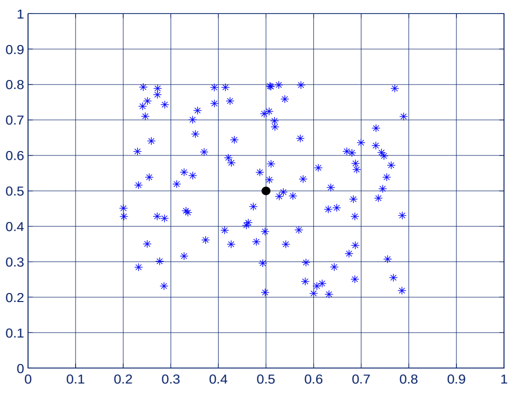
$$rate_{imp} = \frac{\text{the mean of CC-R} - \text{the mean of X}}{max\left(\text{the mean of CC-R}, \text{the mean of X}\right)} \times 100 \tag{1}$$

Where $max\left(\text{the mean of CC-R}, \text{the mean of X}\right)$ is the maximum between the mean of CC-R and the mean of X. Note that the negative values of the $rate_{imp}$ value indicate that the CC-R algorithm can obtain better results than the CC algorithm with the population initialization strategy based on the center point. When $rate_{imp}$ values are positive, it means that the CC algorithm with the population initialization strategy based on the center point archives better results than the CC-R algorithm. Tables II and III shows the improvement values of CC-CNS, CC-CGR, and CC-HRCN algorithms.

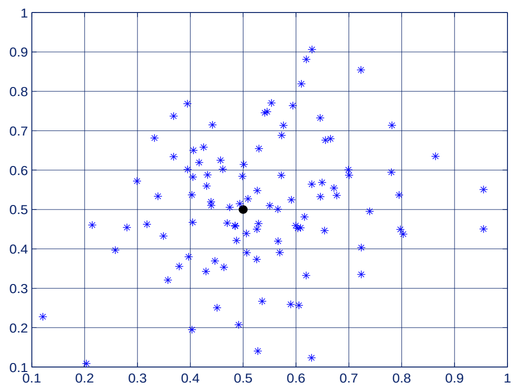
1) Experiment Series 1: Results with the ideal grouping method: Tables I and II show the results of the CC algorithm with the different population initialization strategies along with the ideal grouping method. It is obvious from Table I that CC-CNS performs better than CC-R with NP = 50, 100, and 200 on 6, 8, and 4 out of ten functions, respectively. While CC-R has worse results than CC-R with NP = 50, 100, and 200 on 2 out of ten functions. A closer look at the improvement values from Table II, with NP = 50 and 200, the maximum negative improvement of CC-CNS is less than 7% and the the maximum positive improvement of CC-CNS is greater than 45%. Also, the improvement of CC-CNS is positive on most of functions. With NP=100, CC-CNS was trapped in a local optimum on f_6 and f_{10} and the maximum negative improvement of CC-CNS is very high while on other functions, improvements of CC-CNS are positive. CC-CGR perform better than CC-R with NP = 50, 100, and 200 on 6, 5, and 8 out of ten functions, respectively; but CC-R outperforms CC-CGR with NP = 50, 100, and 200 on 2, 3, and 1 out of ten functions, respectively. From Table II, we can observe that the maximum negative improvement of CC-CGR is less than 9% and the the maximum positive improvement of CC-CGR is greater than 56%.



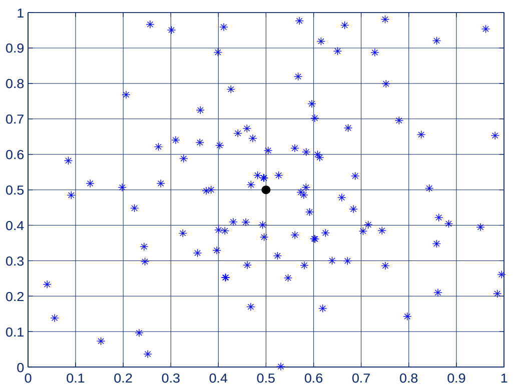
(a) The uniform random initialization strategy



(b) The centeral golden region (CGR) strategy

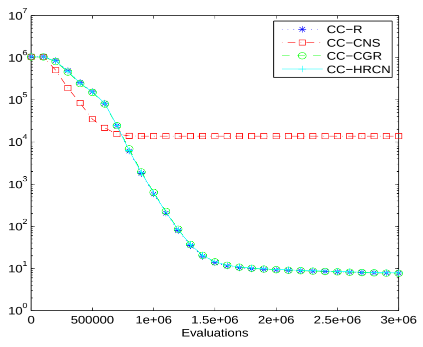


(c) The center-based normal distribution sampling (CNS) strategy

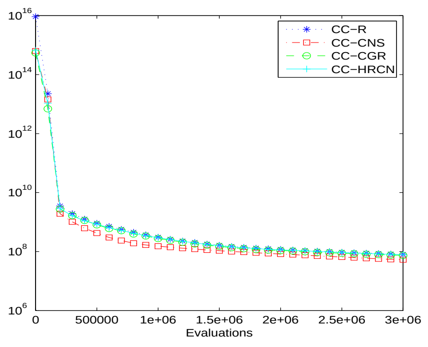


(d) The hybrid random-center normal distribution sampling (HRCN) strategy

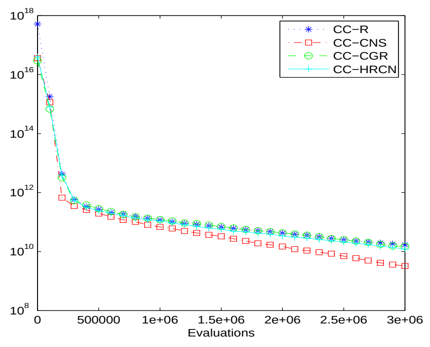
Fig. 3: Four population initialization strategies to generate 100 points in the $2D$ interval of search space as $[-1, 1]^2$.



(a) f_6



(b) f_7



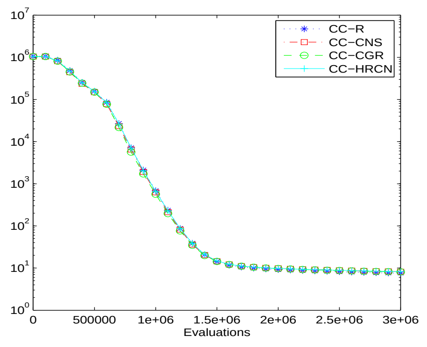
(c) f_{11}

Fig. 4: Convergence plots of f_6, f_7, and f_{11} of the CEC-2013 LSGO benchmark functions. The results were averaged over 25 runs. The vertical axis is the function value and the horizontal axis is the number of function evaluations.

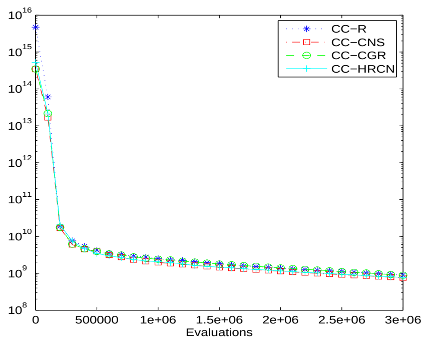
From Table I, CC-HRCN outperforms CC-R with NP = 50, 100, and 200 on 6, 3, and 5 out of ten functions, respectively, while it has worse results than CC-R with NP = 50, 100, and 200 on 2, 3, and 2 out of fifteen functions, respectively. As it can be seen from Table II that the maximum negative improvement of CC-CGR is less than 15% and the the maximum positive improvement of CC-CGR is greater than 39%. From the results, it can be seen that for NP=50, the CC

TABLE I: Results (the mean and the standard deviation of the obtained error values) of CC-CNS, CC-CGR, CC-HRCN, and CC-R algorithms with the ideal grouping method. Symbols '†', '‡', and '≈' denote the compared algorithms are worse than, better than, or similar to CC-R, respectively and the best obtained values are highlighted in bold.

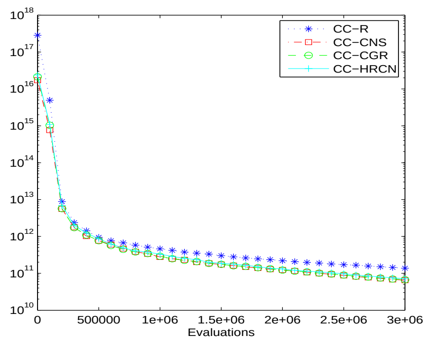
		CC-CNS			CC-CGR			CC-HRCN			CC-R		
Fun	NP	mean	std	median	mean	std	median	mean	std	median	mean	std	median
f_4	50	4.34e+10‡	1.84e+10	3.88e+10	4.64e+10‡	2.04e+10	4.57e+10	**3.94e+10**‡	1.26e+10	4.02e+10	4.97e+10	1.97e+10	4.63e+10
	100	**3.88e+10**‡	1.68e+10	3.63e+10	6.03e+10†	1.47e+10	5.95e+10	7.10e+10†	1.89e+10	7.19e+10	5.58e+10	1.58e+10	5.46e+10
	200	9.28e+10‡	2.56e+10	9.43e+10	**9.24e+10**‡	1.88e+10	8.73e+10	9.61e+10‡	2.50e+10	9.43e+10	1.07e+11	3.33e+10	1.00e+10
f_5	50	5.32e+06‡	3.86e+05	5.35e+06	5.06e+06†	4.21e+05	5.14e+06	5.22e+06†	4.83e+05	5.19e+06	**4.96e+06**	3.63e+05	4.98e+06
	100	**5.23e+06**‡	4.22e+05	5.32e+06	5.84e+06†	3.04e+05	5.73e+06	5.92e+06†	3.14e+05	5.89e+06	5.78e+06	3.10e+05	5.72e+06
	200	6.58e+06†	2.99e+05	6.58e+06	**6.51e+06**‡	3.28e+05	6.50e+06	6.58e+06$^{\approx}$	3.05e+05	6.58e+06	6.55e+06	2.99e+05	6.55e+06
f_6	50	**1.02e+04**$^{\approx}$	2.38e+04	1.49e+01	8.28e+03$^{\approx}$	2.47e+04	1.45e+01	1.41e+04$^{\approx}$	2.73e+04	1.49e+01	1.2e+04	2.51e+04	1.38e+01
	100	1.38e+04†	2.57e+04	1.48e+01	7.68e+00$^{\approx}$	7.56e-01	7.68e+00	**7.86e+00**$^{\approx}$	7.35e-01	7.70e+00	7.75e+00	6.81e-01	7.88e+00
	200	8.04e+00†	4.00e-01	8.11e+00	8.08e+00†	3.63e-01	8.07e+00	8.04e+00†	3.74e-01	8.11e+00	**7.70e+00**	2.67e-01	7.64e+00
f_7	50	**4.81e+07**‡	2.27e+07	4.59e+07	5.99e+07‡	2.08e+07	6.20e+07	5.07e+07‡	2.19e+07	4.38e+07	6.33e+07	2.36e+07	6.67e+07
	100	**5.30e+07**‡	1.95e+07	4.97e+07	7.46e+07‡	1.52e+07	7.27e+07	7.84e+07$^{\approx}$	2.19e+07	7.13e+07	8.04e+07	2.15e+07	7.73e+07
	200	1.17e+08$^{\approx}$	2.63e+07	1.14e+08	1.11e+08‡	1.97e+07	1.10e+08	**1.11e+08**‡	1.85e+07	1.14e+08	1.20e+08	1.88e+07	1.19e+08
f_8	50	5.18e+15†	1.60e+15	4.97e+15	**3.96e+15**‡	1.41e+15	3.84e+15	4.33e+15‡	1.84e+15	3.76e+15	4.86e+15	1.85e+15	5.16e+15
	100	**3.65e+15**‡	1.36e+15	3.26e+15	6.86e+15‡	1.76e+15	6.98e+15	6.98e+15$^{\approx}$	1.59e+15	7.09e+15	7.11e+15	1.93e+15	7.36e+15
	200	1.09e+16‡	2.25e+15	1.10e+16	**1.04e+16**‡	2.44e+15	1.00e+16	1.10e+16‡	2.10e+15	1.10e+16	1.35e+16	2.28e+15	1.29e+16
f_9	50	**4.88e+08**‡	3.23e+07	4.89e+08	4.89e+08‡	3.34e+07	4.89e+08	4.89e+08‡	3.27e+07	4.90e+08	4.97e+08	3.53e+07	4.97e+08
	100	**4.94e+08**‡	3.83e+07	4.99e+08	5.40e+08†	2.38e+07	5.47e+08	5.40e+08$^{\approx}$	3.09e+07	5.51e+08	5.35e+08	2.26e+07	5.33e+08
	200	5.86e+08$^{\approx}$	2.88e+07	5.86e+08	**5.85e+08**$^{\approx}$	2.64e+07	5.90e+08	5.86e+08$^{\approx}$	2.62e+07	5.86e+08	5.87e+08	2.49e+07	5.83e+08
f_{10}	50	8.89e+00‡	1.28e+01	8.05e-01	7.15e+00‡	1.14e+01	8.36e-01	**6.96e+00**‡	1.40e+01	5.35e-01	1.64e+01	1.96e+01	5.28e+00
	100	1.04e+01†	1.65e+01	3.12e+00	**3.48e-02**‡	5.61e-03	3.50e-02	3.73e-02‡	7.58e-03	3.59e-02	3.87e-02	7.02e-03	3.73e-02
	200	**1.93e+02**‡	3.17e+01	1.93e+02	2.10e+02‡	2.97e+01	2.11e+02	2.14e+02‡	2.91e+01	1.93e+02	2.37e+02	2.89e+01	2.35e+02
f_{11}	50	2.15e+09$^{\approx}$	1.55e+09	1.58e+09	2.76e+09$^{\approx}$	2.65e+09	1.61e+09	**1.97e+09**$^{\approx}$	1.08e+09	1.67e+09	3.27e+09	5.11e+09	1.78e+09
	100	**3.25e+09**‡	4.02e+09	3.74e+16	1.49e+10$^{\approx}$	8.51e+09	1.22e+10	1.22e+10‡	6.23e+09	1.10e+10	1.70e+10	1.29e+10	1.30e+10
	200	4.34e+10$^{\approx}$	1.37e+10	4.06e+10	**3.96e+10**‡	1.09e+10	4.14e+10	4.27e+10$^{\approx}$	1.04e+10	4.06e+10	4.36e+10	1.31e+10	4.37e+10
f_{13}	50	**7.93e+09**‡	2.52e+09	7.30e+09	8.59e+09‡	2.33e+09	8.65e+09	9.26e+09‡	2.54e+09	9.08e+09	8.96e+09	2.3e+09	9.02e+09
	100	**7.43e+09**‡	1.52e+09	7.43e+09	1.15e+10‡	2.76e+09	1.16e+10	1.29e+10‡	2.31e+09	1.25e+10	1.21e+10	2.90e+09	1.23e+10
	200	1.56e+10$^{\approx}$	3.06e+09	1.45e+10	**1.40e+10**‡	3.23e+09	1.36e+10	1.61e+10‡	2.94e+09	1.45e+10	1.57e+10	3.24e+09	1.48e+10
f_{14}	50	8.09e+10‡	2.04e+10	7.95e+10	**7.41e+10**‡	1.93e+10	7.27e+10	7.60e+10‡	1.98e+10	6.90e+10	8.77e+10	2.53e+10	8.55e+10
	100	**7.06e+10**‡	1.99e+10	6.79e+10	10.00e+10‡	2.04e+10	9.91e+10	1.02e+11‡	2.01e+10	1.00e+11	1.15e+11	2.82e+10	1.13e+11
	200	**1.33e+11**‡	2.66e+10	1.33e+11	1.36e+11‡	2.75e+10	1.35e+11	1.35e+11‡	2.52e+10	1.33e+11	1.53e+11	4.38e+10	1.39e+11



(a) f_6



(b) f_7



(c) f_{11}

Fig. 5: Convergence plots of f_6, f_7, and f_{11} of the CEC-2013 LSGO benchmark functions. The results were averaged over 25 runs. The vertical axis is the function value and the horizontal axis is the number of function evaluations.

algorithms with three population initialization strategies based on the center point have similar performance. For NP=100, the performances of CC-CNS increases except two functions f_6 and f_{10} while the performances of CC-CGR and CC-HRCN algorithms are deteriorated and their improvements are decreased.

For NP=200, CC-CGR is the best performing algorithm while CC-CGR and CC-HRCN algorithms perform worse than when NP=50 and 100. To gain a better understanding of the behavior of the CC algorithm with the different population initialization strategies along with the ideal grouping method, we plot the convergence graph on three selected problems (f_6-f_7, and f_{11}) with NP=100 in Fig. 4. It can be seen from the Fig. 4, especially the evolutionary processes of functions f_7 and f_{11}, the CC algorithms with the population initialization strategies based on the center point can obtain the better results because of starting with the better solutions. As seen from Fig. 4, the CC algorithms with the population initialization strategies based on the center point can provide the best obtained solution with the lower value in the beginning

TABLE IV: Results (the mean and the standard deviation of the obtained error values) of CC-CNS, CC-CGR, CC-HRCN, and CC-R algorithms with the DG grouping method. Symbols '†', '‡', and '≈' denote the compared algorithms are worse than, better than, or similar to CC-R, respectively and the best obtained values are highlighted in bold.

		CC-CNS			CC-CGR			CC-HRCN			CC-R		
Fun	NP	mean	std	median	mean	std	median	mean	std	median	mean	std	median
f_4	50	**6.03e+10**‡	2.37e+10	5.75e+10	7.07e+10‡	3.43e+10	6.37e+10	8.02e+10$^{\approx}$	3.55e+10	8.13e+10	7.89e+10	3.69e+10	6.46e+10
	100	**1.11e+11**‡	3.65e+10	1.10e+11	1.34e+11‡	6.01e+10	1.25e+11	1.46e+11‡	6.09e+10	1.40e+11	1.58e+11	8.19e+10	1.58e+11
	200	1.68e+11‡	6.64e+10	1.61e+11	**1.40e+11**‡	6.36e+10	1.35e+11	2.14e+11†	8.01e+10	1.90e+11	2.04e+11	8.36e+10	2.11e+11
f_5	50	5.65e+06†	4.84e+05	5.60e+06	**5.46e+06**$^{\approx}$	4.77e+05	5.45e+06	5.66e+06†	4.50e+05	5.71e+06	5.49e+06	3.55e+05	5.43e+06
	100	**6.11e+06**$^{\approx}$	4.06e+05	6.22e+06	6.19e+06$^{\approx}$	3.96e+05	6.15e+06	6.37e+06†	2.49e+05	6.35e+06	6.16e+06	3.21e+05	6.16e+06
	200	7.11e+06†	3.63e+05	7.21e+06	**6.41e+06**‡	3.80e+05	6.40e+06	6.98e+06†	3.00e+05	6.94e+06	6.91e+06	3.34e+05	6.96e+06
f_6	50	1.50e+04$^{\approx}$	2.88e+04	1.51e+01	1.70e+04$^{\approx}$	2.82e+04	1.50e+01	**1.47e+04**$^{\approx}$	2.42e+04	1.52e+01	1.72e+04	2.58e+04	1.49e+01
	100	8.19e+00†	7.38e-01	8.07e+00	8.09e+00†	7.76e-01	7.98e+00	8.23e+00†	8.91e-01	8.27e+00	**7.59e+00**	6.36e-01	7.59e+00
	200	8.10e+00†	3.42e-01	8.05e+00	**7.77e+00**‡	7.26e-01	7.83e+00	8.32e+00†	3.83e-01	8.31e+00	8.03e+00	2.34e-01	8.00e+00
f_7	50	4.84e+08$^{\approx}$	2.03e+08	4.80e+08	**4.15e+08**‡	2.10e+08	3.54e+08	5.09e+08$^{\approx}$	2.94e+08	4.19e+08	5.12e+08	3.08e+08	4.45e+08
	100	**7.53e+08**‡	2.78e+08	7.51e+08	8.85e+08$^{\approx}$	3.84e+08	7.77e+08	7.74e+08‡	3.14e+08	7.08e+08	8.93e+08	3.28e+08	8.93e+08
	200	1.37e+09‡	5.37e+08	1.34e+09	**9.26e+08**‡	3.27e+08	9.34e+08	1.70e+09$^{\approx}$	6.63e+08	1.60e+09	1.60e+09	4.00e+08	1.64e+09
f_8	50	**2.98e+15**‡	1.33e+15	2.99e+15	3.31e+15‡	1.87e+15	3.25e+15	3.95e+15$^{\approx}$	1.87e+15	3.95e+15	4.12e+15	2.02e+15	3.79e+15
	100	**4.08e+15**‡	1.69e+15	3.45e+15	5.18e+15‡	2.12e+15	4.90e+15	5.79e+15‡	2.16e+15	5.89e+15	6.94e+15	3.66e+15	6.94e+15
	200	7.39e+15‡	3.14e+15	6.83e+15	**4.46e+15**‡	1.81e+15	4.38e+15	7.72e+15‡	2.61e+15	8.21e+15	1.01e+16	3.21e+15	9.66e+15
f_9	50	4.90e+08†	2.69e+07	4.95e+08	4.96e+08†	3.61e+07	4.91e+08	**4.77e+08**‡	2.64e+07	4.80e+08	4.84e+08	3.21e+07	4.87e+08
	100	5.33e+08†	2.36e+07	5.38e+08	5.38e+08$^{\approx}$	3.32e+07	5.39e+08	5.40e+08$^{\approx}$	2.53e+07	5.41e+08	**5.37e+08**	2.26e+07	5.37e+08
	200	5.88e+08‡	2.35e+07	5.86e+08	**5.40e+08**‡	2.13e+07	5.44e+08	5.91e+08‡	2.33e+07	5.93e+08	6.00e+08	1.43e+07	6.00e+08
f_{10}	50	**6.40e+00**$^{\approx}$	1.04e+01	1.46e-03	8.54e+00†	1.18e+01	8.05e-01	6.46e+00$^{\approx}$	1.08e+01	5.82e-01	7.17e+00	1.20e+01	5.83e-01
	100	**9.61e-03**‡	2.03e-03	8.90e-03	9.62e-03‡	1.35e-03	9.56e-03	1.00e-02‡	2.29e-03	9.20e-03	1.08e-02	1.92e-03	1.08e-02
	200	4.10e+01‡	4.12e+00	4.07e+01	**9.20e-03**‡	1.22e-03	9.00e-03	4.74e+01‡	5.74e+00	4.75e+01	5.25e+01	4.51e+00	5.32e+01
f_{11}	50	**2.22e+10**‡	1.68e+10	1.58e+10	2.27e+10‡	1.83e+10	1.95e+10	2.89e+10$^{\approx}$	2.29e+10	2.47e+10	4.84e+10	6.63e+10	2.54e+10
	100	**6.50e+10**‡	5.89e+10	4.64e+10	6.70e+10‡	3.53e+10	6.45e+10	7.13e+10‡	7.13e+10	5.00e+10	1.37e+11	1.07e+11	1.37e+11
	200	1.57e+11‡	6.66e+10	1.57e+11	**7.32e+10**‡	5.84e+10	5.94e+10	1.64e+11‡	9.26e+10	1.39e+11	2.06e+11	1.37e+11	1.56e+11
f_{13}	50	1.76e+10‡	4.76e+09	1.68e+10	**1.62e+10**‡	3.31e+09	1.64e+10	1.82e+10‡	4.54e+09	1.74e+10	2.07e+10	6.36e+09	2.11e+10
	100	2.79e+10$^{\approx}$	7.23e+09	2.66e+10	**2.70e+10**‡	5.26e+09	2.58e+10	3.06e+10†	8.10e+09	3.06e+10	2.88e+10	5.18e+09	2.88e+10
	200	4.37e+10†	7.42e+09	4.36e+10	**2.61e+10**‡	4.09e+09	2.60e+10	4.32e+10$^{\approx}$	8.99e+09	4.12e+10	4.30e+10	7.50e+09	4.26e+10
f_{14}	50	**1.62e+10**‡	1.07e+10	1.68e+10	1.68e+10‡	9.82e+09	1.63e+10	1.99e+10‡	1.24e+10	1.93e+10	2.49e+10	1.59e+10	2.45e+10
	100	3.58e+10‡	1.25e+10	3.48e+10	3.75e+10‡	1.32e+10	3.62e+10	**3.84e+10**‡	1.76e+10	3.38e+10	4.39e+10	1.74e+10	4.39e+10
	200	7.83e+10$^{\approx}$	2.30e+10	7.50e+10	**4.19e+10**‡	1.85e+10	3.69e+10	7.51e+10$^{\approx}$	1.68e+10	7.48e+10	8.06e+10	3.35e+10	7.71e+10

TABLE II: $rate_{imp}$ values of CC-CNS, CC-CGR, and CC-HRCN algorithms with the ideal grouping method and the best obtained $rate_{imp}$ values are highlighted in bold.

Function		CC-CNS	CC-CGR	CC-HRCN
f_4	50	12.72%	6.65%	**20.72**%
	100	**30.44**%	-7.50%	-21.41%
	200	13.28%	**13.61**%	10.15%
f_5	50	-6.82%	**-1.99**%	-4.98%
	100	**9.43**%	-1.09%	-2.36%
	200	-0.47%	0.66%	**4.85**%
f_6	50	11.67%	**30.97**%	-14.89%
	100	-99.94%	**0.96**%	-1.40%
	200	-4.30%	-4.74%	**-4.23**%
f_7	50	**24.01**%	5.31%	19.91%
	100	**34.09**%	7.19%	2.49%
	200	2.94%	7.42%	**7.50**%
f_8	50	-6.15%	**18.47**%	10.91%
	100	**48.57**%	3.40%	1.83%
	200	19.50%	**23.12**%	18.52%
f_9	50	**1.86**%	1.68%	1.61%
	100	**7.79**%	-8.64%	-0.93%
	200	0.16%	**0.36**%	0.17%
f_{10}	50	45.77%	56.42%	**57.56**%
	100	-99.63%	**10.00**%	3.75%
	200	**18.61**%	11.43%	9.70%
f_{11}	50	34.36%	15.53%	**39.76**%
	100	**80.88**%	12.39%	28.24%
	200	0.49%	**9.27**%	2.06%
f_{13}	50	**11.51**%	4.18%	-3.24%
	100	**38.51**%	4.88%	-6.20%
	200	0.40%	**11.11**%	-2.48%
f_{14}	50	7.80%	**15.51**%	13.34%
	100	**38.40**%	12.75%	11.30%
	200	**13.06**%	11.08%	11.76%

TABLE III: $rate_{imp}$ values of CC-CNS, CC-CGR, and CC-HRCN algorithms with the DG grouping method and the best obtained $rate_{imp}$ values are highlighted in bold.

Function		CC-CNS	CC-CGR	CC-HRCN
f_4	50	**23.57**%	10.39%	-1.62%
	100	11.39%	**15.19**%	7.59%
	200	17.65%	**31.37**%	-4.67%
f_5	50	-2.83%	**0.55**%	-3.00%
	100	-3.90%	**-0.48**%	-3.30%
	200	-2.81%	**7.24**%	-1.00%
f_6	50	12.79%	1.16%	**14.53**%
	100	**-2.32**%	-6.18%	-7.78%
	200	-0.86%	**3.24**%	-3.49%
f_7	50	5.47%	**18.95**%	0.59%
	100	-3.56%	0.90%	**13.33**%
	200	14.37%	**42.13**%	-5.88%
f_8	50	**27.67**%	19.66%	4.13%
	100	**35.73**%	25.36%	16.57%
	200	26.83%	**55.84**%	23.56%
f_9	50	-1.22%	-2.42%	**1.45**%
	100	-0.56%	**-0.19**%	-0.56%
	200	2.00%	**10.00**%	1.50%
f_{10}	50	**10.74**%	-16.04%	9.90%
	100	**14.81**%	10.93%	7.41%
	200	21.90%	**99.98**%	9.71%
f_{11}	50	**54.13**%	53.10%	40.29%
	100	46.57%	**51.09**%	47.96%
	200	23.79%	**64.47**%	20.39%
f_{13}	50	14.98%	**21.74**%	12.08%
	100	**9.38**%	6.25%	-5.88%
	200	-0.60%	**39.30**%	-0.46%
f_{14}	50	**34.94**%	32.53%	20.08%
	100	4.56%	**14.58**%	12.53%
	200	2.85%	**48.01**%	6.82%

iterations on some functions; f_7 and f_{11}. Note that, the best obtained solutions of all algorithms have different values with the insignificant differences on some functions, e.g., f_6, in the beginning iterations; but it can not be seen a remarkable difference among them due to the scale of plots.

2) Experiment Series 2: Results with the DG grouping method: The results of the CC algorithm with the different population initialization strategies which use the DG grouping method to decompose variables are summarized in Tables III and IV. From Table III, it is obvious that CC-CNS performs better than CC-R with NP = 50, 100, and 200 on 5, 5, and 6 out of ten functions, respectively, while it has worse results than CC-R with NP = 50, 100, and 200 on 2, 3, and 3 out of ten functions, respectively. From Table III, we can observe that the maximum negative improvement of CC-CNS is less than 4% and the the maximum positive improvement of CC-CNS is greater than 46%. From Table III, it is observed that CC-CGR outperforms CC-R with NP = 50, 100, and 200 on 6, 6, and 7 out of ten functions, respectively, while CC-R can better results than CC-CGR with NP = 50, 100, and 200 on 2, 1, and 4 out of fifteen functions, respectively.

As it can be seen from Table IV that the maximum negative improvement of CC-CGR is less than 16% and the the maximum positive improvement of CC-CGR is greater than 32%. It can be seen from Table III, CC-HRCN outperforms CC-R with NP = 50, 100, and 200 on 3, 6, and 5 out of ten functions, respectively, while it has worse results than CC-R with NP = 50, 100, and 200 on 1, 3, and 2 out of fifteen functions, respectively. As it can be seen from Table IV that the maximum negative improvement of CC-CGR is less than 8% and the the maximum positive improvement of CC-CGR is greater than 47%. It can be seen from results, for NP=50, CC-CGR and CC-CNS algorithms behave similar and they have the maximum improvement more than around 50% on function f_{11}.

For NP=100, CC-CGR is the best performing algorithm while for NP=200 both CC-CGR and CC-CNS algorithms have similar performance and are better than CC-HRCN. To gain a better understanding of the behavior of the CC algorithm with the different population initialization strategies along with the DG grouping method, we plot the convergence graph on three selected problems with NP=100 in Fig. 5. As mentioned above, it can be seen from the Fig. 5, especially the evolutionary processes of functions f_7 and f_{11}, the CC algorithms with the population initialization strategies based on the center point can obtain the better results because of starting with the better solutions. In addition, CC-CNS, CC-CGR, CC-HRCN's total number of wins, losses and ties against CC-R algorithm with the DG and ideal decomposition methods are summarized in Table V. From Tables V, it can be observed that the maximum number of wins (8) for ideal decomposition method is obtained by CC-CNS and CC-CGR with NP=100 and 200, respectively. Also, for DG decomposition method the maximum number of wins (10) is obtained by CC-CGR with NP=200. Another observation is that the number of wins by using ideal decomposition method is greater than DG decomposition method on most cases of algorithms, which is expected.

TABLE V: CC-CNS, CC-CGR, CC-HRCN's number of wins, losses and ties against CC-R algorithm. "w/t/l" means that CC-CNS, CC-CGR, CC-HRCN algorithms win in w functions, tie in t functions, and lose in l functions, compared with CC-R algorithm.

			Algorithm		
Decomposition method	NP		CC-CNS	CC-CGR	CC-HRCN
Ideal	50	***w/t/l***	**7/2/1**	**7/2/1**	**7/2/1**
	100	***w/t/l***	**8/0/2**	5/2/3	4/4/2
	200	***w/t/l***	4/4/2	**8/1/1**	6/3/1
DG	50	***w/t/l***	5/3/2	**6/2/2**	3/6/1
	100	***w/t/l***	6/2/2	**6/3/1**	6/1/3
	200	***w/t/l***	6/1/3	**10/0/0**	4/3/3

V. Conclusion

This paper presents an enhanced CC framework with the population initialization strategies based on the center point for handling nonseparable LSGO problems. With help of center-point property, three population initialization strategies, namely, center-based normal distribution sampling (CNS), centeral golden region (CGR), and hybrid random-center normal distribution sampling (HRCN) are employed to the CC framework in order to enhance its performance. These strategies could provide more chance of finding better solutions by using a center point-based sampling method. The center point is a special and crucial point in the search space of a black-box problem with the especial property, the closest point to the unknown-solution. Therefore, using center region in sampling the initial points in the initialization step of population-based algorithms enables them to achieve better results on the nonseparable LSGO problems. The performance of CC algorithms with the population initialization strategies based on the center point was evaluated on CEC-2013 LSGO benchmark functions with D=1000. The experimental results showed that the proposed algorithms are effective and efficient in tackling nonseparable LSGO problems. Furthermore, CC algorithms with the population initialization strategies based on the center region were compared with the CC algorithm with the random initialization strategy (CC-R). The performance of the proposed algorithm is superior to or at least comparable with the CC-R algorithm. Two decomposition methods are applied to CC algorithms, results show that CC algorithms with the center-based strategies perform well by using both decomposition methods, but by using the ideal decomposition, the benefit of the center-based strategies is more highlighted rather than DG method which its performance deteriorates on CEC-2013 LSGO benchmark functions. Although the proposed algorithm has gain some important superiorities on nonseparable LSGO problems, there are still some challenges and difficulties to be further investigated, such as the benefit of center-point property for enhancing the performance of the non-decomposition based algorithms, scalability analysis of algorithms with the population initialization strategies based on the center point, and the effect of population size on these algorithms. Finally, we are interested in investigating the strengths and weaknesses of the center point strategies on separable problems.

References

[1] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from NaturePPSN III*. Springer, 1994, pp. 249–257.

[2] M. A. Potter, "The design and analysis of a computational model of cooperative coevolution," Ph.D. dissertation, Citeseer, 1997.

[3] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Metaheuristics in large-scale global optimization: A survey," *Information Sciences*, 2014.

[4] S. Rahnamayan, H. R. Tizhoosh, and M. M. Salama, "Opposition-based differential evolution," *Evolutionary Computation, IEEE Transactions on*, vol. 12, no. 1, pp. 64–79, 2008.

[5] S. Rahnamayan and G. G. Wang, "Center-based sampling for population-based algorithms," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009, pp. 933–938.

[6] A. Esmailzadeh and S. Rahnamayan, "Enhanced differential evolution using center-based sampling," in *Evolutionary Computation (CEC), 2011 IEEE Congress on*. IEEE, 2011, pp. 2641–2648.

[7] A. Esmailzadeh and S. Rahnamayan, "Center-point-based simulated annealing," in *Electrical & Computer Engineering (CCECE), 2012 25th IEEE Canadian Conference on*. IEEE, 2012, pp. 1–4.

[8] X. Li, K. Tang, M. N. Omidvar, Z. Yang, K. Qin, and H. China, "Benchmark functions for the cec 2013 special session and competition on large-scale global optimization," *gene*, vol. 7, p. 33, 2013.

[9] Y. Liu, X. Yao, Q. Zhao, and T. Higuchi, "Scaling up fast evolutionary programming with cooperative coevolution," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 2. IEEE, 2001, pp. 1101–1108.

[10] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. IEEE, 2008, pp. 1663–1670.

[11] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.

[12] H. K. Singh and T. Ray, "Divide and conquer in coevolution: A difficult balancing act," in *Agent-Based Evolutionary Search*. Springer, 2010, pp. 117–138.

[13] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in *Parallel Problem Solving from Nature, PPSN XI*. Springer, 2010, pp. 300–309.

[14] Y. Ren and Y. Wu, "An efficient algorithm for high-dimensional function optimization," *Soft Computing*, vol. 17, no. 6, pp. 995–1004, 2013.

[15] J. Liu and K. Tang, "Scaling up covariance matrix adaptation evolution strategy using cooperative coevolution," in *Intelligent Data Engineering and Automated Learning–IDEAL 2013*. Springer, 2013, pp. 350–357.

[16] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 18, no. 3, pp. 378–393, June 2014.

[17] E. Sayed, D. Essam, and R. Sarker, "Using hybrid dependency identification with a memetic algorithm for large scale optimization problems," in *Simulated Evolution and Learning*. Springer, 2012, pp. 168–177.

[18] E. Sayed, D. Essam, and R. Sarker, "Dependency identification technique for large scale optimization problems," in *Evolutionary Computation (CEC), 2012 IEEE Congress on*. IEEE, 2012, pp. 1–8.

[19] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Cooperative co-evolution with a new decomposition method for large-scale optimization," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 1285–1292.

[20] M. N. Omidvar, Y. Mei, and X. Li, "Effective decomposition of large-scale separable continuous functions for cooperative co-evolutionary algorithms," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 1305 – 1312.

[21] Z. Ma and G. A. Vandenbosch, "Impact of random number generators on the performance of particle swarm optimization in antenna design," in *Antennas and Propagation (EUCAP), 2012 6th European Conference on*. IEEE, 2012, pp. 925–929.

[22] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. IEEE, 2008, pp. 1110–1116.

[23] S. Shan and G. G. Wang, "Metamodeling for high dimensional simulation-based design problems," *Journal of Mechanical Design*, vol. 132, no. 5, p. 051009, 2010.

[24] M. N. Omidvar, X. Li, and X. Yao, "Smart use of computational resources based on contribution for cooperative co-evolutionary algorithms," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 2011, pp. 1115–1122.

[25] L. Sun, S. Yoshida, X. Cheng, and Y. Liang, "A cooperative particle swarm optimizer with statistical variable interdependence learning," *Information Sciences*, vol. 186, no. 1, pp. 20–39, 2012.

[26] T. Ray and X. Yao, "A cooperative coevolutionary algorithm with correlation based adaptive variable partitioning," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009, pp. 983–989.