Competitive Two-Island Cooperative Coevolution for Real Parameter Global Optimisation

Rohitash Chandra and Kavitesh Bali
School of Computing Information and Mathematical Sciences
University of the South Pacific, Suva, Fiji.
Email: c.rohitash@gmail.com, Email: bali.kavitesh@gmail.com

***Abstract*—Cooperative coevolution has proven to be efficient in solving global optimisation and real world application problems. However, it is highly sensitive to problem decomposition, especially in the context of non-separable functions that possess interacting decision variables. Problem decomposition has been a challenge of cooperative coevolution. Efficient problem decomposition strategy ensures that interacting variables are grouped into separate subcomponents. Introduction of competition and collaboration features have shown to be advantageous in evolutionary algorithms but have not quite been fully explored in cooperative coevolution. In this paper, a method is utilized that enforces competition in coevolution whereby different problem decomposition schemes are implemented as islands that compete and collaborate with each other. The proposed framework is tested on several global optimisation benchmark problems and achieves promising results.**

I. Introduction

Cooperative coevolution is an evolutionary computation method which solves a problem by dividing it into smaller subcomponents [1]. A major characteristic of this is the ability to simplify the complexities of a problem through decomposition [2]. Cooperative coevolution has been applied to real parameter global optimisation problems including high dimension problems [3], [4], [5], [6]. Cooperative coevolution has also been used for neuro-evolution [7], [8], [9] for a wide range of problems including time series prediction and pattern classification [10], [2], [11].

Problem decomposition is a major issue in cooperative coevolution that affects its performance as inter-dependencies are present among decision variables in non-separable problems [12], [1]. The grouping of interacting variables into separate subcomponents has been the challenge of cooperative coevolution. A good problem decomposition method groups variables with inter-dependencies together [12]. A number of strategies in the past have been used for problem decomposition where interacting variables have been grouped heuristically [3], [4], [5], [6].

Cooperative coevolution naturally retains diversity through the use of sub-populations, where mating is restricted to the sub-populations and cooperation is anticipated mainly through collaborative fitness evaluation [1], [7]. Since selection and recombination is restricted to a sub-population, the new solution will basically not inherit features from the rest of the subpopulations. Therefore, cooperative coevolution produces more diverse population when compared to a standard evolutionary algorithm with a single population [7]. Competition is a major feature in biological evolution. The initial motivations for using competition in evolutionary algorithms was presented in a competitive coevolution method where a population called "host" and another called "parasite" competed with each other with different mechanisms that enabled fitness sharing, elitism and selection [13]. In cooperative coevolution, competition has been used for multi-objective optimisation [14] that exploited correlation and inter-dependencies between the components of the problem. Competition has also been used in cooperative coevolution based multi-objective optimisation in dynamic environments where problem decomposition method adapts according to the change of environment rather than being static from the beginning of the evolution [15].

In cooperative coevolution for training recurrent neural networks, competition has been enforced through different problem decomposition methods that decomposed the neural network and implemented as islands [16]. Competition can ensure that the different problem decomposition methods are given an opportunity during the entire evolution and there is not a problem in finding the right problem decomposition method at a particular time according to the degree of separability [2] .

This paper applies competitive island based cooperative coevolution [16], [17] to real parameter global optimisation problems. We propose a two island competition model that incorporates different problem decomposition in terms of number and size of subcomponents. The performances of this algorithm is measured with *fixed size* (uniform) problem decomposition as well as *arbitrary* problem decomposition strategies. The proposed method is applied to problems with 60 and 100 dimensions and compared to standalone cooperative cevolution with the same problem decomposition schemes that were used as islands.

The rest of the paper is organized as follows. In Section II, a background of cooperative coevolution with recent advancement in tackling the issue of problem decomposition and separability has been given. Section III provides details of the proposed framework and its application to selected global optimisation problems. Section IV discusses the experimental results and Section V concludes the paper with discussion of future works.

978-1-4799-7492-4/15/$31.00 ©2015 IEEE

II. Background: Problem Decomposition in Cooperative Coevolution

Problem decomposition is considered to be a core aspect of cooperative coevolution. The problem decomposition method solely relies on the nature of the optimisation problem. The original cooperative coevolution framework (CCEA) decomposed the problem in a way where a single sub-population is used for each variable [1]. The sub-populations in the cooperative coevolution framework are evolved separately and cooperation takes place for fitness evaluation for the respective individuals in each sub-population. The general framework for function optimisation is given in Algorithm 1 which outlines how the large problem is decomposed. The algorithm begins by initialising and cooperatively evaluating each of individuals of the respective sub-populations. After the initialisation and evaluation phase, the evolution proceeds. All the sub-populations are evolved in a round-robin fashion for the *depth* of n generations. A *cycle* is complete when all the sub-populations have been evolved for n generations. The algorithm terminates until the maximum number of cycles are reached or the minimum error is satisfied. The size of a subcomponent and the way a subcomponent is encoded is dependent on the problem.

Algorithm 1: The General Cooperative Coevolution Framework

```
1) Decompose the problem into $k$ subcomponents
2) Initialise and cooperatively evaluate each subcomponent
represented as a sub-population
while until termination do
  for each Subpopulation do
    for $n$ Generations do
      i) Select and build new individuals
      ii) Cooperatively evaluate the new individuals
      iii) Update sub-population
    end for
  end for
end while
```

In order to use cooperative coevolution for optimisation problems that fall between total separable and fully non-separable, it is important to group interacting variables within a subcomponent. One of the other foremost problems researchers try to address with cooperative coevolution is the decomposition of main problem into subcomponents. The grouping of interacting variables into the separate subcomponents could minimise interaction between subcomponents and a large partially non-separable problem can then get tackled as separable problem that can take full advantage of cooperative coevolution.

In an ideal decomposition, the parameters are grouped into subcomponents such that the inter-dependencies between variables are kept at minimum [6]. Fast evolutionary programming in the cooperative co-evolutionary framework (FEPCC) has been the first attempt to tackle function optimisation of up to 1000 dimensions [18]. FEPCC used the original cooperative coevolution evolutionary algorithm (CCEA) framework by Potter and Jong [1]. The major drawback of CCEA is that it does not have the mechanism to provide interaction between subcomponents which is needed for non-separable problems. Due to this, FEPCC performed poorly on non-separable problems.

Cooperative Coevolution based Particle Swarm optimisation (CPSO) [19], unlike the original cooperative coevolution framework [1], decomposes the problem into m s-dimensional subcomponents where s is the number of variables in a subcomponent. Shi *et. al* presented a Differential Evolution based cooperative coevolution framework which divides the problem into halves and each half is optimized using Differential Evolution [3]. The method was applied for large-scale problems of only 100 dimensions. This strategy does not work for problems of high dimension as the halved subcomponents cannot cope with higher dimensions. Yang *et. al* [4] have presented the cooperative coevolution framework that uses a *random grouping* and *adaptive weighting* strategy with differential evolution (DECC-G) for its subcomponents. The method groups interacting and non-interacting variables into separate subcomponents heuristically. The framework was used for non-separable problems of up to 1000 dimensions. Omidvar et. al. [6] made amendments to the adapting weighting approach in DECC-G. They argued that the *random grouping* approach in DECC-G proved to capture two interacting variables in a subcomponent with a probability which gets significantly lower when more than two interacting variables are present in the problem.

An extension to the CPSO has been the cooperative coevolution framework for particle swarm optimisation (CCPSO) [20] which has been applied to large-scale non-separable problems using random grouping and adapting weighting present in DECC-G [4]. CCPSO showed to perform significantly better for large scale problems of up to 1000 dimensions than its differential evolution counterpart in DECC-G. The authors concluded that cooperative coevolution can be a good method for tackling the limitations of particle swarm optimisation in handling problems of high dimensions.

Ray and Yao presented a cooperative coevolution framework using correlation based adaptive partitioning technique (CCEA-AVP) [21]. The authors of [21] provided insights into why CCEA in its basic form was not suitable for non-separable problems and introduced (CCEA-AVP) to deal with those problems. This offered the possibility to deal with problems where separability among variables might vary in different regions of search space. In this method, the problem is evolved for a few generations using a single population and then the correlation coefficients of the top 50 % of the individuals of the population are calculated.

Chen *et. al* presented the cooperative coevolution with variable interaction learning (CCVIL) [22] which divides the optimisation problem into two stages. These are the learning stage and the optimisation stage that execute once in sequence. In the learning stage, all decision variables are treated independently and cooperatively evolved from which the interacting variables are identified using variable interaction learning and variables are merged into groups. Authors of [22] proclaim that CCVIL is significantly better than DECC-G and MLCC (two state-of-the-art CC-based algorithms) and the internal optimizer JADE. However, the effectiveness of CCVIL within real-world application domain is still unknown [22]. Omidvar *et. al* presented another approach to large scale non-separable problems using *Delta Grouping* [23]. This method computes the averaged difference of a certain variable across the entire population which is used for identifying interacting variables. The method is based on the idea presented by Salomon [12] who showed that coordination rotation is one way of turning a separable problem into an non-separable one.

Yang *et. al* presented a multi-level cooperative coevolution framework (MLCC) [5] which adapts the size of the sub-components in DECC-G in order to group interacting and non-interacting variables. The framework starts with small sized subcomponents and adapts to bigger subcomponents sizes from a predefined set. MLCC showed better performance than DECC-G for non-separable problems of up to 1000 dimensions. In a recent study, Omidvar [24] recommended that the *rule of thumb* for selecting a subcomponent size is to choose it small enough so that it is within the capacity of the subcomponent optimizer,The fact that the capacity of an optimizer is not always known before-hand , and finding the optimal subcomponent size requires elaborate empirical studies and experiments [24], a viable solution is to adapt the subcomponent size during the optimization. He then proposed an adaptive method, (MLSoft) [24], that uses widely-used techniques in reinforcement learning to adapt the subcomponent size during the optimization process. The experimental results suggest that MLSoft is significantly better than the former existing adaptive algorithm (MLCC).

III. Competitive Island-Based Cooperative Coevolution - CICC

The proposed method (CICC) employs the strength of different problem decomposition strategies that reflect on the different degree of non-separability (interaction of variables) [12], [2] and diversity (number of sub-populations) during evolution [1], [7].

In this section, we propose a cooperative coevolution method that incorporates competition and collaboration with species that is motivated by evolution in nature. The proposed method, namely, *Competitive Island-Based Cooperative Co-evolution - CICC,* employs different problem decomposition strategies that compete and collaborate via islands. Different islands enable competition by comparing their solutions after a fixed time in terms of number of fitness evaluations and exchange the best solution between the islands. In this paper, we focus on two island competition model that can be extended to multiple islands later.

A. Initialisation

To anticipate competition between the two islands, both islands begin search with the *same* random values in a range, but are divided into different Problem Decomposition (PD) strategies which defines the number of subcomponents and the size of each respective subcomponent. The subcomponents can be of equal dimensions and can also be of varied sized dimensions. The subcomponents are implemented as sub-populations.

We refer to a problem decomposition configuration as an *island* in the proposed competitive two-island cooperative coevolution method. We ensure that both islands (Island One and Island Two) begin with the same genetic materials in the population, but evolve them differently as defined by the different problem decomposition methods. Initially, all the sub-populations of Island One are initialised with random-real number values from a range. Next, these real values (from Island One) are copied into the sub-populations of Island Two which has a different problem decomposition strategy.

A different problem decomposition configuration is employed for each island which is defined by the number and size of each subcomponent. A problem decomposition configuration can either have same sized or varied sized subcomponents. The highest level of decomposition for an island would have one subcomponent for each variable. We can only evolve each island for complete cycles and therefore, the number of function evaluations cannot be exactly the same for each island. The number of function evaluation depends on the number of sub-populations used in the island as an island with greater number of subcomponents will cost more function evaluations to evolve for each cycle. Different islands adapt to different times (FEs) because the search difficulties along different dimensions of each island are different. The islands basically compete and collaborate with each other until it reaches the maximum number of FEś set to 1.5e+06 or when there is no improvement identified for 5000 FEś in the second half of the evolutionary process as described below in section IV.

B. Competition

Algorithm 2 explains the mechanism behind the proposed *Competitive Island Based Cooperative Coevolution* in detail. In Stage 1, the sub-populations of Island One and Island Two with their respective problem decomposition strategies are cooperatively evaluated. Stage 2 proceeds with evolution in an island based round-robin fashion where each island is evolved in isolation for a predefined time. This is called *island evolution time* that is given by the number of *cycles* that makes the required number of function evaluations for the respective islands. A cycle is complete when all the sub-populations have been evolved for n number of generations in a round-robin fashion.

Once a particular island has been evolved for the island evolution time, the algorithm proceeds and checks if the best solution (fitness) of that particular island is better than the rest of the islands. Afterwards, the *collaboration* phase takes place, whereby the best identified solution is copied to the other island. In this manner, the best solution is used to help and improve the other island compete fairly in the next round. During the course of the Collaboration procedure, the algorithm needs to take into account how the best identified solution from one island will be transferred into the other island as shown in Figure 1.

C. Collaboration

A number of factors needs to be taken into account when making a transfer as the size and number of subcomponents vary for each island due to their different problem decomposition strategies. The island that contains an individual with better solution is then injected (copied) into the other islands as outlined in Stage 4 of Algorithm 1. The best individuals from each of the subcomponents needs to be carefully concatenated into an individual and transferred without losing any genotype (subcomponents in cooperative coevolution) to phenotype (recurrent neural network) mapping.

TABLE I. PROBLEM DEFINITIONS BASED ON [25], [26], [27]

Problem	Name	Optimum	Range	Multi-modal	Fully Separable	Error
F1	Ellipsoid	0	[-5,5]	No	Yes	1E-20
F2	Shifted Sphere	-450	[-100,100]	No	Yes	1E-10
F3	Schwefel's Problem 1.2	0	[-5,5]	No	Yes	1E-20
F4	Rosenbrock	0	[-5,5]	Yes	No	1E-20
F5	Shifted Rosenbrock	390	[-100,100]	Yes	No	1E-10
F6	Rastrigin	0	[-5,5]	Yes	Yes	1E-20
F7	Shifted Rastrigin	-330	[-5,5]	Yes	Yes	1E-10
F8	Shifted Griewank	-180	[-600,600]	Yes	No	1E-10

Algorithm 2: Competitive Two-Island Cooperative Co-evolution

```
Stage 1: Initialisation:
i. Cooperatively evaluate Island One
ii. Cooperatively evaluate Island Two
Stage 2: Evolution:
while FuncEval $\leq$ GlobalEvolutionTime do
    while FuncEval $\leq$ Island-Evolution-Time do
        foreach Sub-population at Island-One do
            foreach Depth of $n$ Generations do
                Create new individuals using genetic operators
                Cooperative Evaluation of Island One
            end
        end
    end
    while FuncEval $\leq$ Island-Evolution-Time do
        foreach Sub-population at Island-Two do
            foreach Depth of $n$ Generations do
                Create new individuals using genetic operators
                Cooperative Evaluation of Island Two
            end
        end
    end
    Stage 3: Competition: Compare and mark the island with best fitness.
    Stage 4: Collaboration: Inject the best individual from the island with
    better fitness into the other island.
    if ErrorIslandOne $\leq$ ErrorIslandTwo then
        Copy Island One's best individual into the Island Two.
    end
    else
        Copy Island Two's best individual into Island One.
    end
end
```

The winner island is used to inject the best solution to the other island. The island in which the best individual is injected is evaluated to ensure that the injected individual has a fitness. In order to save evaluation time, the fitness can also be transferred along with the solution. This depends on the way the sub-populations are implemented and the approach taken in ensuring that the fitness value is updated at the right position that corresponds with the individual that has been transferred. Since each sub-population contain individuals that have a fitness, we need to note that there will be a number of different fitness values from the best individual in each sub-population. We only take the best fitness value and use it to replace the best individuals from all the sub-populations in the other islands as shown in Figure 2. The best individuals from one island is transferred into the other. In the reverse case, the fitness of the last individual of Island Two is copied to each of the best individuals of Island One. The size and number of sub-populations can be different and therefore, only the best fitness replace the old best fitness as it carries a stronger solution.

IV. EXPERIMENTS & RESULTS

This section shows the performance of the proposed method on selected global optimization problems of 60 and 100 dimensions. For this paper, the eight benchmark functions have been selected considering the level of difficulty, the scope of separability and the nature of problem, i.e. unimodal or

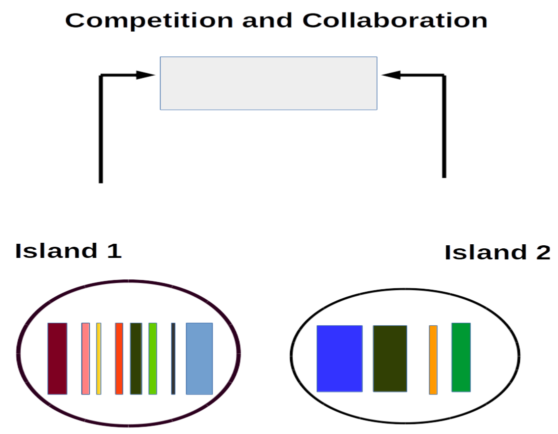

Fig. 1. Competition and Collaboration in Competitive Cooperative Coevolution. The best solution competes within the rest of the solutions from same island until the local evolution time has been reached.

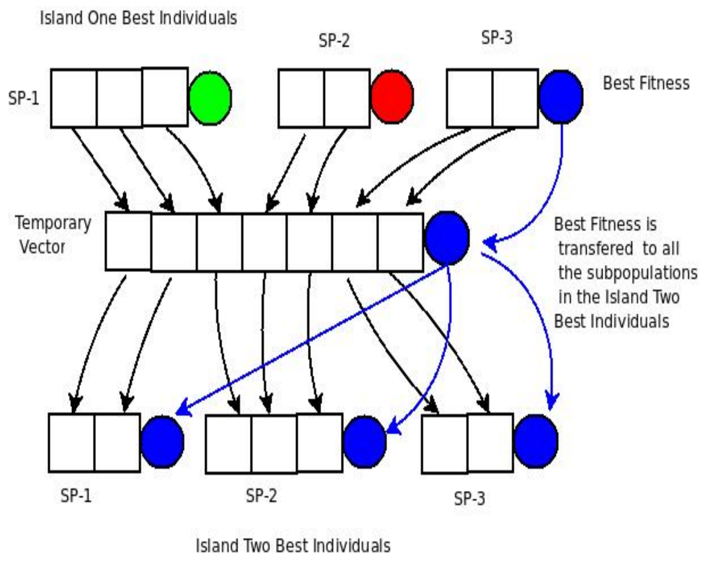
Island One Best Individuals
SP-1
SP-2
SP-3
Best Fitness
Temporary Vector
Best Fitness is transfered to all the subpopulations in the Island Two Best Individuals
SP-1
SP-2
SP-3
Island Two Best Individuals

Fig. 2. Individuals shown as square box is copied from Island One into Island Two. A single best fitness from one of the individuals (shown as circles) is copied to all the individuals in Island Two. Note that only the fitness of the last individual is copied as this fitness is the main fitness of Island One.

TABLE II. UNIFORM PROBLEM DECOMPOSITION

Problem Decomp.	Dimensions	Num. SP	Size SP
PD1	60	4	15
PD2	60	10	6
PD3	60	15	4
PD1	100	10	10
PD2	100	4	25
PD3	100	20	5

multimodal listed by Table I.

The generalised generation gap with parent-centric crossover evolutionary algorithm (G3-PCX) [28] is used in all the sub-populations of cooperative coevolution. We use a pool size of 2 parents and 2 offspring as used in [28] . A population size of 1000 is used for all experiments as it performed better in trial experiments and helps avoid premature convergence for fairly higher dimensions of up to 100.

In the competitive island based cooperative coevolution, we use different problem decomposition strategies where the same subcomponent size for each sub-population (SP) was used as shown in Table II. According to [24], these values that represent low, medium and high dimensional subcomponent sizes allow us to approximately determine the optimal subcomponent size. Next, we also test this algorithm by arbitrarily dividing the problem dimensions into relevant subcomponents as such in Table VI. The global optimisation functions that were selected had features in terms of being fully-separable, non-separable, unimodel and multi-modal as shown in Table I. This enables us to examine if the proposed method is suitable in a wide range of problems and we can also highlight its limitations. The column that marks "Error" in Table I determines one of the stopping criteria (i.e. LIMIT) before the algorithm reaches maximum number of function evaluation which is fixed as 1500 000. A *successful* run is only when the algorithm halts with the minimum error. The other stopping criteria is when no improvement is made for 5000 function evaluations in the second half of the evolutionary process which also defines an unsuccessful run. The results for 50 experimental runs with 95% confidence interval are provided in Tables III-VIII. In each case the mean fitness value (error) has been reported along with the success rates. Results are presented in the following subsection.

A. Results

The first set of experiments were conducted for problems with 60 dimensions and the results are shown in Table III. The results show that the proposed competitive cooperative coevolution method (PD1-PD2) have performed fairly well in terms of number of function evaluations and success rate for F1 - F5. The proposed approach has improved the unimodal and fully separable problems (F1-F3) and two multi-modal and partially separable problems (F4 and F5). Furthermore, this approach has not been able to perform quite well for (F6-F7) which are difficult multi-modal and non-separable problems in terms of function evaluations. However, In both cases of F6 and F7, though they could not converge to the optimum values, the proposed competitive method has achieved better fitness (lower error) when compared to their stand alone counter-parts, problem decomposition one (PD1) and problem decomposition two (PD2). The fact that G3-PCX is local search intensive,

TABLE III. THE RESULTS THAT COMPARE CICC WITH CC FOR 60 DIMENSIONS

PD	Prob.	FE	Error	Success/50
PD1	F1	328446 ± 1248	9.33E-21 ± 1.39E-22	50
PD2	F1	204271 ± 646	9.47E-21 ± 1.17E-22	50
PD1-PD2	F1	159048 ± 375	8.04E-21 ± 3.12E-22	50
PD1	F2	196884± 1042	-450 ± 1.76E-12	50
PD2	F2	122920 ± 398	-450 ± 1.18E-12	50
PD1-PD2	F2	96456 ± 309	-450 ± 3.80E-12	50
PD1	F3	592836 ± 1221	2.58E-20 ± 2.67E-20	47
PD2	F3	342854 ± 864	4.92E-21 ± 8.80E-22	50
PD1-PD2	F3	169524 ± 1213	1.37E-21 ± 7.12E-22	50
PD1	F4	1204968 ± 2097	0.55 ± 0.38	43
PD2	F4	744621 ± 1630	0.31 ± 0.29	46
PD1-PD2	F4	323532 ± 2111	0.23 ± 0.26	47
PD1	F5	1138212 ± 2134	390.71 ± 0.42	41
PD2	F5	675513 ±1346	390.63 ± 0.40	42
PD1-PD2	F5	271716 ± 523	390.23 ± 0.26	47
PD1	F6	756300 ± 1097	134.14 ± 5.92	0
PD2	F6	750120 ± 0	197.45 ± 7.99	0
PD1-PD2	F6	750600 ± 0	105.48 ± 5.03	0
PD1	F7	750300 ± 239	-196.37 ± 6.53	0
PD2	F7	750120 ± 433	-92.36 ± 11.04	0
PD1-PD2	F7	752664 ± 1488	-216.79 ± 4.67	0
PD1	F8	562206 ± 1297	-179.99 ± 2.66E-03	17
PD2	F8	387626 ± 1787	-179.99± 1.71E-03	29
PD1-PD2	F8	584604 ± 1054	-179.99 ± 6.31E-03	30

TABLE IV. THE RESULTS THAT COMPARE CICC WITH CC FOR 100 DIMENSIONS

PD	Prob.	FE	Error	Success/50
PD1	F1	448680 ± 1097	9.58E-21 ± 8.37E-23	50
PD2	F1	362145 ± 1451	9.71E-21 ± 5.93E-23	50
PD3	F1	692280 ± 923	9.46E-21 ± 1.12E-22	50
PD1-PD2	F1	365208 ± 1438	8.81E-21 ± 2.01E-22	50
PD1-PD3	F1	392520 ± 651	7.96E-21 ± 3.77E-22	50
PD2-PD3	F1	165696 ± 386	8.20E-21 ± 3.93E-22	50
PD1	F2	272958 ± 1422	-450 ± 1.09E-12	50
PD2	F2	209500 ± 771	-450 ± 9.06E-13	50
PD3	F2	416472 ± 1571	-450 ± 1.56E-12	50
PD1-PD2	F2	212664 ± 631	-450 ± 2.12E-12	50
PD1-PD3	F2	233760 ± 721	-450 ± 3.13E-12	50
PD2-PD3	F2	101760 ± 248	-450 ± 2.76E-12	50
PD1	F3	707154 ± 341	3.20E-19 ± 4.40E-19	47
PD2	F3	454456 ± 388	5.52E-21 ± 1.08E-21	49
PD3	F3	954372 ± 205	7.14E-15 ± 8.58E-15	16
PD1-PD2	F3	414888 ± 197	4.74E-21 ± 8.44E-22	50
PD1-PD3	F3	361200 ± 221	9.48E-22 ± 6.20E-22	50
PD2-PD3	F3	200376 ± 1338	1.00E-21 ± 6.58E-22	50
PD1	F4	1500000 ± 0	17.27 ± 0.88	0
PD2	F4	1451296 ± 342	0.71 ± 0.42	9
PD3	F4	1500000 ± 0	50.18 ± 3.01	0
PD1-PD2	F4	1008720 ± 709	0.39 ± 0.33	27
PD1-PD3	F4	1481220 ± 198	10.15 ± 4.05	1
PD2-PD3	F4	986568 ± 1800	0.08 ± 0.15	23

the issue of premature convergence could be addressed by introducing adaptive diversity mechanisms and more importantly, finding a better problem decomposition to the problem. A better analysis of different problem decomposition strategies has been conducted in the case of 100 dimensions.

The results for the second set of problems with 100 dimensions is shown in Table IV and Table V. We use three different competitive setups as we employ three different problem decomposition methods for this case (PD1, PD2 and PD3). This basically lets us gauge the performances of different problem decomposition strategies and possibly determine the optimal decomposition based on the generalisation in [24].

The results show the performance of the proposed competitive cooperative coevolution method given by combinations: (PD1-PD2), (PD2-PD3) and (PD1-PD3).

In general, the competition enforced by PD2-PD3 has given

TABLE V. THE RESULTS THAT COMPARE CICC WITH CC FOR 100 DIMENSIONS

PD	Prob.	FE	Error	Success/50
PD1	F5	1489320 ± 1342	409.26 ± 1.78	3
PD2	F5	1456464 ± 656	390.95 ± 0.47	6
PD3	F5	1500000 ± 0	446.33 ± 3.60	0
PD1-PD2	F5	880104 ± 1482	390.71 ± 0.42	33
PD1-PD3	F5	1304220 ± 1321	397.97 ± 2.19	13
PD2-PD3	F5	766464 ± 1211	390.08 ± 0.15	43
PD1	F6	750300 ± 0	271.68 ± 8.67	0
PD2	F6	750120 ± 0	388.98 ± 14.49	0
PD3	F6	771000 ± 1298	191.98 ± 8.11	0
PD1-PD2	F6	750600 ± 0	392.66 ± 15.13	0
PD1-PD3	F6	766080 ± 141	155.11 ± 4.94	0
PD2-PD3	F6	762408 ± 1116	158.13 ± 6.45	0
PD1	F7	750300 ± 0	-36.68 ± 13.11	0
PD2	F7	750120 ± 0	101.69 ± 18.93	0
PD3	F7	760200 ± 309	-140.38 ± 7.48	0
PD1-PD2	F7	750600 ± 231	122.03 ± 15.73	0
PD1-PD3	F7	757560 ± 133	-177.35 ± 5.34	0
PD2-PD3	F7	751776 ± 1117	-152.86 ± 7.39	0
PD1	F8	433458 ± 1876	-179.99 ± 1.59E-03	33
PD2	F8	406975 ± 767	-179.99 ± 1.56E-03	32
PD3	F8	573240 ± 1652	-179.99 ± 2.91E-03	26
PD1-PD2	F8	426168 ± 1264	-179.99 ± 1.58E-03	32
PD1-PD3	F8	438000 ± 1211	-179.97 ± 1.97E-02	30
PD2-PD3	F8	255888 ± 889	-179.99 ± 1.993E-03	44

the best performance for all the problems when compared to the rest of the competitive combinations and standalone problem decomposition methods (PD1, PD2, PD3). PD1-PD3 and PD1- PD2 have not made major improvements for F1-F3 problems which is different from the results given in Table III for 60 dimensions. In comparision to the other problem decompositions, The pair (PD2-PD3) showed promising results when tested on the multimodal, non-separable Rosenbrock Function, F4. Since, the 100D combination of (PD2-PD3) had the better performance overall interms of minimum fitness and minimum FE's, we propose that this method performs better for problems with higher dimensions.

In Table V, the competitive methods have generally performed better for problems F5-F8 which are the more difficult problems. This is in terms of the fitness, function evaluations and success rate. In problems F6 and F7, PD2-PD3 has shown better results according to the fitness of the problem. Due to the nature of the problem, the local search-intensive G3-PCX optimizer could not converge to the global optimum and therefore, all the methods showed no success in general for these problems. With a fine success rate in problem F8, we can deduce that the problem decomposition combination (PD2-PD3) was the optimal and that an effective decompostion should be as such that the subcomponent size is neither too small nor too large [24]. This is however dependent on the performance of the subcomponent optimizer.

B. Arbitrary Problem Decomposition

As it is difficult to know what might be the effective and accurate problem decomposition until evolution, it is important to experiment the proposed algorithm with other problem decomposition mechanisms such as arbitrary division or problem decomposition. In such cases, the subcomponents are compromised of different (non-uniform) dimension sizes. Thus, we test the CICC with a combination of two different arbitrary sets sizes which makes total of 100 dimensions as presented in Table VI. It was noteworthy that the proposed algorithm outperformed the standalone cooperative coevolution (PD1, PD2 and PD3) in most test functions.

TABLE VI. ARBITRARY PROBLEM DECOMPOSITION FOR 100 DIMENSIONS

Problem Decomp.	Decomposition Strategy
PD1	[8,12,11,9,7,13,14,6,13, 7]
PD2	[10,12,8,9,8,15,7,11,9, 11]

TABLE VII. THE RESULTS THAT COMPARE CICC WITH CC FOR 100 DIMENSIONS - ARBITRARY PROBLEM DECOMPOSITION

PD	Prob.	FE	Error	Success/50
PD1	F1	156085 ± 712	9.67E-21 ± 7.91E-23	50
PD2	F1	362145 ± 1451	9.71E-21 ± 5.93E-23	50
PD1-PD2	F1	115860 ± 1719	4.81E-21 ± 7.20E-22	50
PD1	F2	22684 301 ± 301	-450 ± 1.09E-12	50
PD2	F2	22476 ± 246	-450 ± 1.07E-12	50
PD1-PD2	F2	11820 ± 1132	-450 ± 5.45E-12	50
PD1	F3	136160 ± 770	5.81E-21 ± 7.92723E-22	50
PD2	F3	140890 ± 556	5.52E-21 ± 8.15594E-22	50
PD1-PD2	F3	42060 ± 197	1.33E-21 ± 5.79E-22	50
PD1	F4	500040 ± 0	60.41 ± 4.45	0
PD2	F4	500040 ± 0	58.80 ± 1.59	0
PD1-PD2	F4	1169820 ± 1733	27.51 ± 8.83	13

TABLE VIII. THE RESULTS THAT COMPARE CICC WITH CC FOR 100 DIMENSIONS - ARBITRARY PROBLEM DECOMPOSITION

PD	Prob.	FE	Error	Success/50
PD1	F5	500040 ± 0	452.26 ± 4.75	0
PD2	F5	500040 ± 0	457.359 ± 6.72	0
PD2-PD3	F5	1207200 ± 1334	433.56 ± 8.90	5
PD1	F6	500040 ± 0	341.35 ± 12.27	0
PD2	F6	500040 ± 0	336.22 ± 12.93	0
PD1-PD2	F6	753240 ± 0	325.97 ± 10.65	0
PD1	F7	490417 ± 1987	108.86 ± 16.08	1
PD2	F7	500040 ± 0	103.94 ± 12.22	0
PD1-PD2	F7	753000 ± 0	59.22 ± 12.52	0
PD1	F8	433458 ± 1876	-179.99 ± 1.59E-03	33
PD2	F8	406975 ± 767	-179.99 ± 1.56E-03	32
PD1-PD2	F8	185280 ± 1303	-180.00 ± 1.77E-03	39

Even with arbitrary problem decomposition, the island based competition mechanism (PD1-PD2) scored a success rate of 13 (F4) and 5 (F5), respectively. It outweighs the zero success scores of standalone PD1 and PD2 and converged to a better solution.

Despite the nil success rate for F6, the PD1-PD2 combination still had the lowest (better) error recorded alongside minimum function evaluations in comparison to PD1 and PD2. In terms of minimum function evaluations and error rate, F7 recorded a similar performance presenting a lower error value. The competitive combination (PD1-PD2) worked well for F8 as well. Results are presented in Tables VII and VIII.

C. Further Comparisons

To determine the feasibility of the proposed Competitive Two Island Cooperative Coevolution (CICC), we conduct a comparison on the selected benchmark problems from some of the algorithms in literature that took part in the 2008 Congress on Evolutionary Computation Competition [25]. These include Multilevel Cooperative Coevolution [5] MLCC) , Dynamic Multi-Swarm Particle Swarm Optimizer [29] (DM-SPSO) and Self-Adaptive Differential Evolution algorithm [30] (jDEdynNP-F).

The median, mean and the standard deviation of the function errors *(f(x)-f(*x))* of 25 runs are presented with respect to their max function evaluations of 5e+5 for 100 dimensions. Statistical data is proposed in Table IX for a comparative analysis.

In general, CICC has shown promising performance on the tested set with respect the other algorithms. It achieved significantly promising results on 3 out 4 of the functions being compared. These include Shifted Sphere, Shifted Rosenbrock and Shifted Griewank.

Along side the three algorithms in comparision, CICC performed fairly well as the rest of the algorithms on Shifted Sphere and Shifted Griewank function . Additionally, CICC outperformed all the rest of the algorithms on the multimodal, non-separable Shifted Rosenbrock function. JDedynNP-F and MLCC outperformed CICC on Shifted Rastrigin function and generally performed equally well as CICC on the rest of the functions in terms of the selected performance metrics, such as the median, mean, and standard deviation of the objective function values. Due to the local search intensive property of G3-PCX, CCIC did not quite perform well on the multimodal yet separable Shifted Rastrin function, yet it did outperform DMSPSO and recorded a smaller function error value for the same. Generally, we can speculate that this Competitive Island based Cooperative Coevolution works well with different types of functions with dimensions of upto 100D. This motivates a scale up separate study of CICC on large scale global optimisation problems for upto 1000 Dimensions. We leave this as a next future plan.

D. Discussion

Competition and collaboration are important features of evolutionary algorithms. The results have showed that the proposed method can be helpful for cooperative coevolution as it is difficult to establish the right problem decomposition strategy. Arbitrary problem decomposition and competition during evolution has shown to be quite effective. Cooperative coevolution enables greater diversity and helps in global search when higher level of decomposition is present in terms of higher number of subcomponents. This is helpful in the beginning but can be a problem when interacting variables are not grouped in separate subcomponents. The proposed competitive island based cooperative coevolution ensures that it retains diversity and global search and also deals with issue of separability through the problem decomposition strategies that compete and collaborate with each other. By following the general rule of thumb described in [24], an optimal user defined problem decomposition can be attained. Basically, an effective subcomponent size is neither too small, nor too large and is dependent on the performance of the optimiser. The results have shown that different problem decomposition strategies yield different performances and vary for different types of problems. Therefore, it is important to have different combinations for the two islands. This gives motivation to implement more islands in the competition. In future, three or more number of islands can be used in the competition. Howsoever, the computation time in terms of transferring the best solution also has to be taken into account. When more islands are assigned to the CICC model for competition, the winner island injects its solution to the rest of the islands after a certain time, and help motivate them to compete in the next round. With respect to the solution injection (migration) phase, this CICC framework has been able to improve evolutionary requirements such as *diversity* during evolution. The performance of the sub-populations with a lower diversity is revived during the migration of individuals from one island to the other.

It should be noted that the basic goal of this paper is real-parameter global optimisation and not specifically Large scal global optimisation. In future, we are interested in doing a scale up study of the proposed CICC model by applying it to higher dimensional (1000D) benchmark functions proposed in [26] and measure the efficiency of the proposed model by comparing it with the state of the art algorithms such as Differential Grouping [31].

V. Conclusion

In this paper, a competitive island-based cooperative coevolution was applied to real parameter global optimisation problems. The proposed approach employed two island competitive methods that were defined by different problem decomposition methods in terms of the size and number of subcomponents.

The results show that in most cases, the proposed competition cooperative coevolution method outperformed the standalone problem decomposition methods in terms of function evaluations and success rate. The results are promising when compared to some of the established methods from the literature. This gives motivation for further research whereby the same ideas can be utilized in implementing more islands with different problem decomposition methods, and can also be extended to different evolutionary algorithms in the sub-populations.

In future work, efficient strategies, such as adaptive problem decomposition methods can be incorporated to boost the performance of the proposed island based competition method. The optimisation time can be decreased with multi-threaded implementation. This proposed framework can be applied to real world application problems and also for combinatorial optimisations.

Acknowledgment

The authors would like to thank Mr. Nabi Omidvar, a phd candidate at RMIT, Melbourne, Australia, for providing relevent feedback and future ideas to this proposed Competitive Island based Cooperative Coevolution model.

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] R. Chandra, M. Frean, and M. Zhang, "On the issue of separability for problem decomposition in cooperative neuro-evolution," *Neurocomputing*, vol. 87, pp. 33–40, 2012.

[3] Y.-j. Shi, H.-f. Teng, and Z.-q. Li, "Cooperative co-evolutionary differential evolution for function optimization," in *Advances in natural computation*. Springer, 2005, pp. 1080–1088.

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

[5] ——, "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.

TABLE IX. COMPARISON BETWEEN CICC, MLCC[5], DMSPSO[29], JDEDYNNP-F[30] FOR 100D.

Functions	Stats.	Algorithms			
		MLCC [5]	DMSPSO [29]	jDEdynNP-F [30]	**CICC** (PD2-PD3)
Shifted Sphere	Median	5.68e-14	0.00e+00	5.68e-14	8.15e-13
	Mean	6.82e-14	0.00e+00	5.68e-14	8.25e-13
	StDev	2.32e-14	0.00e+00	0.00e+00	2.93e-14
Shifted Rosenbrock	Median	1.42e+02	2.58e+02	1.13e+02	8.12e-02
	Mean	1.50e+02	2.83e+02	1.15e+02	8.00e-02
	StDev	5.71e+01	9.40e+02	4.47e+01	7.65e-02
Shifted Rastrigin	Median	4.55e-13	1.83e+02	5.68e-14	1.67e+02
	Mean	4.34e-13	1.83e+02	5.46e-14	1.72e+02
	StDev	9.21e-14	2.16e+01	1.14e-14	1.03e+01
Shifted Griewank	Median	2.84e-14	0.00e+00	2.84e-14	8.67e-14
	Mean	3.41e-14	0.00e+00	2.84e-14	8.53e-14
	StDev	1.16e-14	0.00e+00	0.00e+00	1.43e-14

[6] M. N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative co-evolution for large scale optimization through more frequent random grouping," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*. IEEE, 2010, pp. 1–8.

[7] M. A. Potter and K. A. De Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary computation*, vol. 8, no. 1, pp. 1–29, 2000.

[8] N. García-Pedrajas and D. Ortiz-Boyer, "A cooperative constructive method for neural networks for pattern recognition," *Pattern Recognition*, vol. 40, no. 1, pp. 80–98, 2007.

[9] F. Gomez and R. Miikkulainen, "Incremental evolution of complex general behavior," *Adaptive Behavior*, vol. 5, no. 3-4, pp. 317–342, 1997.

[10] F. Gomez, J. Schmidhuber, and R. Miikkulainen, "Accelerated neural evolution through cooperatively coevolved synapses," *The Journal of Machine Learning Research*, vol. 9, pp. 937–965, 2008.

[11] R. Chandra and M. Zhang, "Cooperative coevolution of elman recurrent neural networks for chaotic time series prediction," *Neurocomputing*, vol. 86, pp. 116–123, 2012.

[12] R. Salomon, "Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. a survey of some theoretical and practical aspects of genetic algorithms," *BioSystems*, vol. 39, no. 3, pp. 263–278, 1996.

[13] C. D. Rosin and R. K. Belew, "New methods for competitive coevolution," *Evolutionary Computation*, vol. 5, no. 1, pp. 1–29, 1997.

[14] C. K. Goh, K. C. Tan, D. Liu, and S. C. Chiam, "A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design," *European Journal of Operational Research*, vol. 202, no. 1, pp. 42–54, 2010.

[15] C.-K. Goh and K. Chen Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 13, no. 1, pp. 103–127, 2009.

[16] R. Chandra, "Competition and collaboration in cooperative coevolution of Elman recurrent neural networks for time-series prediction," *Neural Networks and Learning Systems, IEEE Transactions on*, p. In Press, 2015.

[17] ——, "Competitive two-island cooperative coevolution for training elman recurrent networks for time series prediction," in *Neural Networks (IJCNN), 2014 International Joint Conference on*. IEEE, 2014, pp. 565–572.

[18] 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.

[19] F. Van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 8, no. 3, pp. 225–239, 2004.

[20] X. Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009, pp. 1546–1553.

[21] T. Ray and X. Yao, "A cooperative coevolutionary algorithm with correlation based adaptive variable partitioning," in *Proceedings of the Eleventh conference on Congress on Evolutionary Computation*, ser. CEC'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 983–989. [Online]. Available: http://portal.acm.org/citation.cfm?id=1689599.1689729

[22] 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.

[23] M. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping for large scale non-separable function optimization," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, 2010, pp. 1762 –1779.

[24] 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.

[25] K. Tang, X. Yao, P. Suganthan, C. MacNish, Y. Chen, C. Chen, and Z. Yang, "Benchmark functions for the CEC'2008 special session and competition on large scale global optimization," Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep., 2007. [Online]. Available: http://nical.ustc.edu.cn/cec08ss.php

[26] 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," RMIT University, Melbourne, Australia, Tech. Rep., 2013. [Online]. Available: http://goanna.cs.rmit.edu.au/ xiaodong/cec13-lsgo

[27] F. Herrera, M. Lozano, and D. Molina, "Test suite for the special issue of soft computing on scalability of evolutionary algorithms and other metaheuristics for large scale continuous optimization problems," Tech. Rep., 2010. [Online]. Available: http://sci2s.ugr.es/eamhco/functions1-21.pdf

[28] K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization," *Evolutionary computation*, vol. 10, no. 4, pp. 371–395, 2002.

[29] S.-Z. Zhao, J. J. Liang, P. N. Suganthan, and M. F. Tasgetiren, "Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. IEEE, 2008, pp. 3845–3852.

[30] J. Brest, A. Zamuda, B. Boskovic, M. S. Maucec, and V. Zumer, "High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. IEEE, 2008, pp. 2032–2039.

[31] 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, 2014.