On the Relationship of Degree of Separability with Depth of Evolution in Decomposition for Cooperative Coevolution

Rohitash Chandra* Ratneel Deo† Kavitesh Bali† and Anurag Sharma‡
*†Artificial Intelligence and Cybernetics Research Group
Software Foundation, Nausori, Fiji
http://aicrg.softwarefoundationfiji.org
Email: c.rohitash@gmail.com
†‡School of Computing Information and Mathematical Sciences
University of South Pacific, Suva, Fiji.
Email: deo.ratneel@gmail.com

***Abstract*—Problem decomposition determines how subcomponents are created that have a vital role in the performance of cooperative coevolution. Cooperative coevolution naturally appeals to fully separable problems that have low interaction amongst subcomponents. The interaction amongst subcomponents is defined by the degree of separability. Typically, in cooperative coevolution, each subcomponent is implemented as a sub-population that is evolved in a round-robin fashion for a specified depth of evolution. This paper examines the relationship between the depth of evolution and degree of separability for different types of global optimisation problems. The results show that the depth of evolution is an important attribute that affects the performance of cooperative coevolution and can be used to ascertain the nature of the problem in terms of the degree of separability.**

I. Introduction

Despite their successful applications [1, 2], the performance of evolutionary algorithms (EA's) depreciates with an increase in the number of dimensions [3, 4]. This is also known as the "curse of dimensionality"[3]. The reasons appear to be two-fold [5]; 1) complexity of the problem tends to increase with the size of problem, where a previously successful search strategy may not necessarily be capable of finding the optimal solution in a different context. 2) the solution space of the problem increases exponentially with respect to the problem size. There is a need to have enhanced search strategies to explore all the promising regions in a given time budget [6, 7, 8, 5].

Cooperative Coevolution (CC) is a biologically inspired divide and conquer evolutionary algorithm [6] that features subcomponents and evolves them as sub-populations. The subcomponents typically consist of a single or a group of interacting variables which evolve together in isolation. Although CC has been a promising framework for confronting large dimensional optimisation problems, only a few studies have reported performance effects of decomposing a high dimensional problem into single or low dimensional subcomponents [7, 9]. CC has been applied to neuro-evolution for solving a wide range of time-series [10, 11] and pattern classification problems [12]. Problem decomposition has been widely been studied in case of neuro-evolution where decomposition relies on the network architecture and degree of separability for different decomposition methods have been experimentally studied [12].

CC in principle, performs best in problems that can be broken down and that do not have any inter-dependencies or interaction within the decision variables [13]. Such functions are referred to as separable functions that feature independent decision variables and thus can be easily decomposed. Non-separable function on the contrary feature inter-dependencies amongst the decision variables and therefore are harder to decompose [14]. Most of the research done with CC has been focusing on decomposing the problem into smaller sub-components as problem decomposition strategies are dependent on the degree of separability of the problem [15, 9] .

There has been a lot of work done with regards to problem decompositions [15, 9], however, not much research has been done regarding the depth of evolution for different types of problems. The depth of evolution is basically the number of generations a sub-population is evolved. Similarly, observations have not been done on the behavior of problem separability based on depth of evolution.

This paper experimentally examines the relationship between the depth of evolution and degree of separability for different types of global optimisation problems. The aim is to establish a relationship between depth of evolution and separability and to find how it affects the performance on the different types of problems that for instance are fully or partially separable. This can further help us understand the relationship of separability and diversity in global optimisation problems. Hence, the aim can be fulfilled with the following research questions.

- Does the depth of evolution play an important role for different types of problems in terms of separability?

- Does the nature of problem change in terms of separability at different stages of evolution?
- Is the degree of separability the major factor for performance of cooperative coevolution?

In answering the research questions, the experiments are designed to feature basic cooperative coevolution for the selected benchmark functions. The methodology is motivated by a study of separability in previous work in case of neuro-evolution where the variables were perturbed and the change in gradient of the neural network was observed [12]. In a fully separable problem, a deep search of evolution will not create any issues in terms of local convergence as solving a variable independently means that you are partially solving the problem.

The rest of this paper is organized as follows. A brief overview on separability and modality is given in *Section II*, and *Section III* gives details of the methodology. *Section IV* presents the experiments followed by the results, and discussion. *Section VI* concludes the paper with directions for future research.

II. Background and Related Work

A. Global Optimization

Optimization is a process that aims to find the maximum or minimum value out of all possible solutions to a problem. Global optimisation problems deal with various types of functions ranging from fully separable, fully non-separable and partially separable functions [16, 17, 18]. These functions are grouped into two classes based on modality [19]. Unimodal functions are those functions that have only one global minimum and does not have local minimums. A multi-modal function basically has one or more local minimums along with the global minimum. Unimodal functions are easier to optimize because of its lack of local minimums while multi-modal functions are tougher as there are chances that the search gets stuck at a local minimum. The optimisation problem gets more complex as the number of decision variables and their interactions escalate [20].

B. Cooperative Coevolution

Cooperative Coevolution has demonstrated to be very promising in solving global optimisation problems [6]. Different variants of cooperative coevolution have been utilized where the issue of problem decomposition and separability has been central [7]. The performance of CC algorithms on high dimensional problems could be significantly enhanced by incorporating more advanced EAs. Yang and Yao [7], proposed a cooperative co-evolutionary framework to address high dimensional non-separable problems. They utilized adaptive weighting to permit co-adaptation among subcomponents while they are interdependent. They also proposed a group based problem decomposition strategy whereby the grouping structure could be changed dynamically. Their framework performed fairly well with large scale problems partaking interacting variables. Omidvar et. al [15] proposed a cooperative co-evolutionary differential evolution to discourse high dimensional problems of up to 1000 dimensions which showed promising results. Their goal was to advance an earlier algorithm [21] by employing principal component analysis to condense the dimensions of a problem [22].

More recently, Chandra et. al presented an adaptive method known as competitive island based cooperative coevolution (CICC)[23] where candidate solutions were grouped into islands that compete and collaborate. The best individual from the winning islands is injected into the losing island to ensure fair competition in different phases of evolution for global optimisation [23]. The same method was earlier used for training Elman recurrent neural networks for time series prediction [24] with promising results. Omidvar et. al [25] presented dependent variables into subcomponents based on the deferential grouping method and achieved improvements in the problem decomposition strategies employed.

III. Methodology: Analysis of Depth of Evolution in Cooperative Coevolution for Separability

In this section, we define separability and discuss how cooperative coevolution algorithm can be used to study the relationship between depth of evolution and separability.

Algorithm 1: Cooperative Coevolution

```
Stage 1: Initialization:
i. Randomly Generate a set of individuals and assign
them into the sub-populations randomly
ii. Evaluate Species
Stage 2: Evolution:
while Func-Eval ≤ Global-Evolution-Time do
    foreach Sub-population do
        foreach Depth of n Generations do
            Create new individuals using genetic
            operators
            Cooperative Evaluation the population
            Update the sub-populations
        end
    end
end
```

The canonical cooperative coevolution algorithm employs a divide and conquer scheme where the population is divided into sub-populations of various dimensions. The subcomponents are implemented as sub-populations as shown in Figure 1. The type of problem decomposition determines the number and size of each of the subcomponents. In the beginning of evolution, each sub-population contains individuals that have arbitrary fitness which is evaluated cooperatively. Cooperative evaluation is done with representative examples where typically the best individual from each of the sub-populations are concatenated with the current individual and evaluated using the fitness function.

Although any evolutionary algorithm can be used in the sub-populations, we use the generalized generation gap with parent-centric crossover (G3-PCX) evolutionary algorithm [26].

A. Depth of Evolution vs Separability

The depth of evolution is defined as the number generations that each sub-population is evolved in a round robin fashion. A cycle in cooperative coevolution is complete when all the sub-populations have been evolved. This is executed until the number of fitness evaluations or when the minimum error or fitness is attained.

Definition 1:
A function of n variables is separable if it can be written as a sum of n functions with just one variable as given in Equation 1. The parameters of a separable function are called independent variables [27].

$$\underset{(x_1,x_2,..,x_n)}{\arg\min}\ f(x_1,x_2,..,x_n) = \left(\underset{x_1}{\arg\min}\, f(x_1,..),..\underset{x_n}{\arg\min}\, f(..x_n)\right) \tag{1}$$

The Quadratic function is an example of a separable problem given by Equation 2.

$$\min_x f(\mathbf{x}) = \sum_{i=1}^{n} i x_i^4 + \mathcal{N}(0,1) \tag{2}$$

$$-100 \leq x \leq 100, x^* = (0,0,..,0)\ and\ f(x^*) = 0$$

In real world problems, interdependencies exist among decision variables. Problems that have interdependencies between decision variables are commonly termed as non-separable. In [18, 17], the degree of separability have been further divided into two classes as given in Definition 2. These are *m-non-separable* and *fully-non-separable*.

Definition 2: A non-separable function $f(x)$ is called a m-non-separable function if at most m of its parameters x_i are not independent. A non-separable function $f(x)$ is called fully-non-separable function if any two of its parameters x_i are not independent [17, 18] .
An example of a non-separable problem is the extended Rosenbrock function as given in Equation 3. Examples of large-scale separable and non-separable functions with up to 1000 dimensions have been provided in [16, 17, 18].

$$\min_x f(x) = \sum_{i=1}^{n-1}[(1-{x_i}^2) + 100(x_{i+1} - x_i^2)] \tag{3}$$

The extended Rosenbrock function has been shown to have exactly 1 minimum for n=3 at (1,1,1) and exactly 2 minima for $4 \leq n \leq 7$. This result has been obtained by setting the gradient of the function equal to zero [28].

Due to the existence of variable interactions in non-separable problems, optimizing each variable independently may hinder the convergence to a high quality solution. This highlights the hurdle and challenge to solve fully non-separable problems. Seemingly, most problems are *partially separable* [29, 30], that is they fall between the two extremes of fully separable and fully non-separable [29, 30, 18].

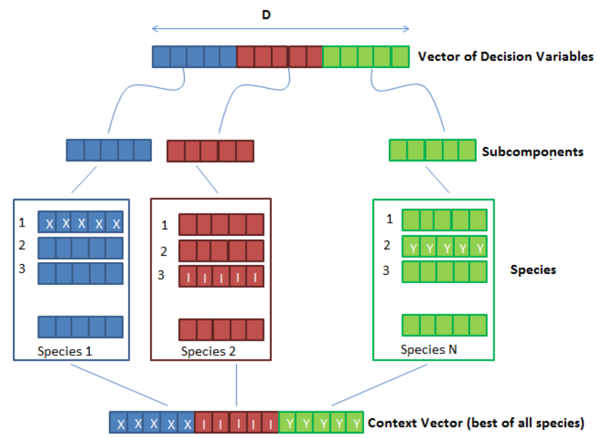


Fig. 1. Decomposing a large dimension problem into smaller sub components. Each sub-component represents a species and the combinations of the best individuals from these species forms the context vector

B. Methodological Design

We used three experimental design strategies to investigate the research questions.

- The relationship of between the degree of separability and the depth of evolution is established by observing the performance of the algorithm by varying the depth of evolution.
- In order to evaluate if degree of separability changes during evolution, we evaluate the performance at different depths of evolution at the selected stages of the optimisation process.
- The effect of the degree of separability is based on observation of the performance of each problem decomposition strategy evaluated on different benchmark problems.

IV. Experiments and Analysis

In this section, we present experimental analysis for the relationship of depth of evolution with degree of separability. We choose benchmark problems in global optimisation that have distinct qualities in terms of fitness landscape that reflect separability and multi-modality. We then present details of experimental design where different problem decomposition methods are tested for each problem where the depth of evolution is evaluated.

There were three classes of functions that we studied; fully separable unimodal [*f1, f2 and f3*], non-separable multi-modal [*f4, f5 and f8*] and non-separable uni-modal [*f6 and f7*]. Table I gives a description of the benchmark problems that have different fitness landscapes in terms of separability and multi-modality. The minimum error that defines the termination criteria is also given. Each problem takes different termination condition in terms of the number of function evaluations as given in Table I. The experiments were designed such that the algorithm terminates when the minimum error is reached within the maximum time frame for each problem.

We observed the performance of the algorithm on the different benchmark problems for depth of evolution (DE) that

TABLE I
PROBLEM DEFINITIONS BASED ON [31, 32, 33]

Problem	Name	Optimum	Range	Multi-modal	Fully-Separable	Error	Maximum-Time(FE)
f1	Ellipsoid	0	[-5,5]	No	Yes	1E-20	15000
f2	Shifted Sphere	-450	[-100,100]	No	Yes	1E-10	10000
f3	Schwefel's Problem 1.2	0	[-5,5]	No	Yes	1E-20	20000
f4	Rosenbrock	0	[-5,5]	Yes	No	1E-20	500000
f5	Shifted Rosenbrock	390	[-100,100]	Yes	No	1E-10	500000
f6	Rastrigin	0	[-5,5]	Yes	Yes	1E-20	500000
f7	Shifted Rastrigin	-330	[-5,5]	Yes	Yes	1E-10	500000
f8	Shifted Griewank	-180	[-600,600]	Yes	No	1E-10	500000

TABLE II
PROBLEMS DECOMPOSITION STRATEGIES FOR DECOMPOSING 100 DIMENSION PROBLEM

Decomposition	Number of Subcomponents	Subcomponent Size
PD1	4	25
PD2	25	4
PD3	10	10
PD4	5	20
PD5	20	5

was varied from 1 generation to 100 generations in increments of 20 generations. The main focus of the investigation was about finding the relationship between the depth of evolution and the error in the problems stated above. The algorithm was analyzed for 100 dimensions with different problem decomposition methods that are summarized in Table II. Note that the problems were decomposed by adjacent variables. All the sub-populations contained 100 individuals. The G3-PCX evolutionary algorithm was used to evolve the sub-populations with parameters that included 2 parents and 2 offspring. We use the cooperative coevolution implementation from the *Smart Bilo Computational Intelligence Framework* [34].

We show the performance of each problem decomposition at the different depth of evolution for two cases in the optimisation process. We show the results in terms of performance of the algorithm for different problem decomposition methods at the initial 10% of maximum evolution time and then at 100% convergence. A termination criterion was also when there were no changes in the minimal change of error for x function evaluations. We chose x as 100 based on results from initial runs of the experiments.

V. RESULTS AND DISCUSSION

A. Results

The results are given in the Figure 2(a) to Figure 9(b) where the error of each function (problem) is affected as DE is varied at two different stages in the optimisation process. Figure 2(a) to 9(a) show the performance for DE vs error at 10% convergence while Figure 2(b) to 9(b) show the performance at 100% convergence.

We observed that the Ellipsoidal function [*f1*] was fully converged at approximately 15 000 function evaluations. The different PD's and depths of evolution did not make any difference when we look at the final result after optimisation given in Figure 2(b). Contrarily, in the initial phases, the depth and PD played a significant role in affecting the error. The different PD's had different function error as seen in Figure 3(a). Similarly, for the depth of evolution, the smaller depth of evolution had given a larger error that gradually decreased as the depth of evolution increased.

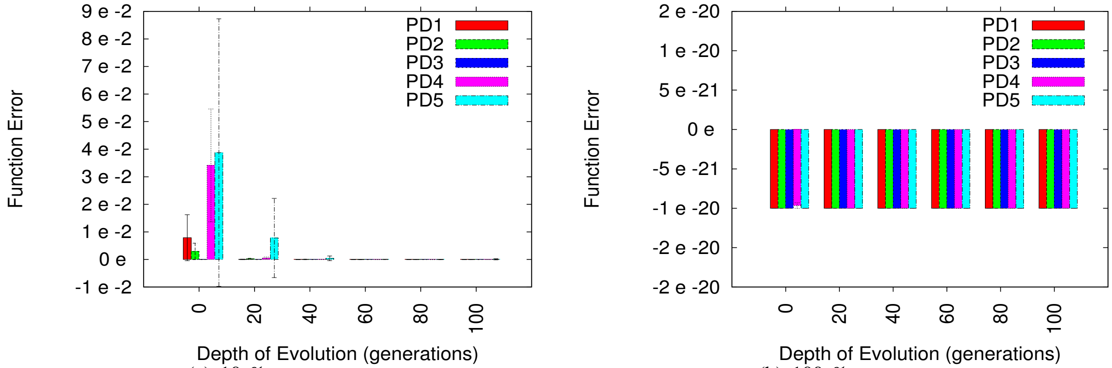
The Shifted Sphere [*f2*] and Schwefel's Problem 1.2 [*f3*], Figures 2 and 3, followed a similar pattern as *f1*. The depth of evolution had an influence on the different PD's at 10 % of optimisation time. At 100% optimisation time, the depth of evolution did not make a large impact on the error, especially for more than 20 generations. The PD, on the contrary, was seen to be greatly affecting the error for a smaller depth of evolution (less than 30 generations) throughout the optimisation process. For instance, in Figure 4(a) and 4(b), textitPD4 is chaotic at both 10% and 100% optimisation time for lower depth of evolution. Note that we refer to 10 % of optimisation time as the beginning of evolution and 100 % as the end of evolution in the rest of the discussion.

The Rosenbrock [*f4*], Shifted Rosenbrock [*f5*] and Griewank function [*f8*] seem to have similar trend as the previous functions (*f1*,*f2* and *f3*). The DE at the beginning of evolution influences the error whereas the lower DE produced a higher error. In the case of Figure 5 , at the beginning of evolution, *PD2* reduces the error as the DE increases. The Shifted Rosenbrock function also clearly shows that the PD behaves differently at different stages of optimisation. Figure 5 shows that *PD2* is performing as the poorest at the beginning, however, at the end of optimisation, PD4 reported the poorest performance.

The Rastrigin and Shifted Rastrigin functions (*f7* and *f7*), showed no affinity towards the varying depth of evolution throughout the optimisation process. Figure 6 and 7 show that neither the PD nor the DE affect these functions at the beginning and similarly at the end.

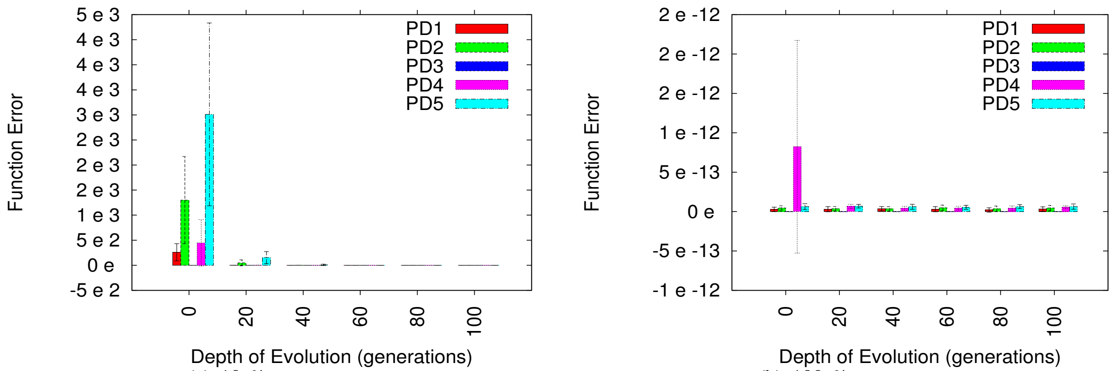
B. Discussion

In the results, we found that the performance of cooperative coevolution produced similar trend for different values in the depth of evolution for fully separable uni-modal functions and non-separable multi-modal functions. These problems



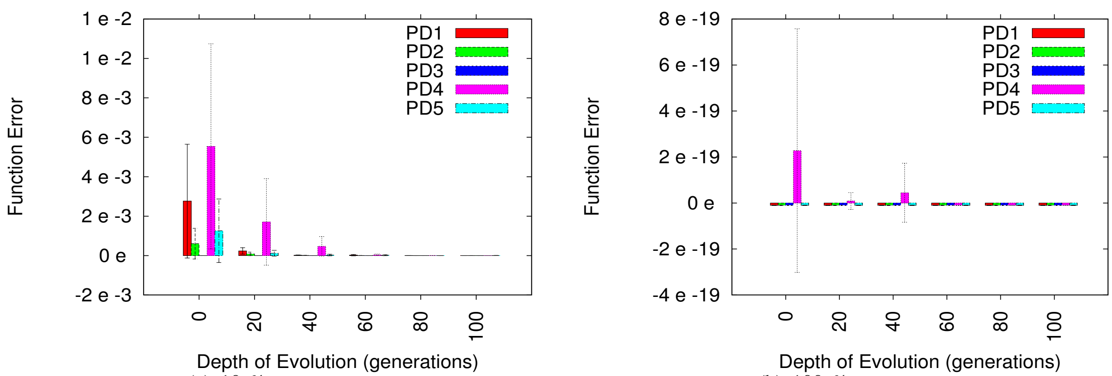
(a) 10 % convergence
(b) 100 % convergence

Fig. 2. Depth of evolution vs error for Ellipsoid function [*f1*] at different stages of convergence.



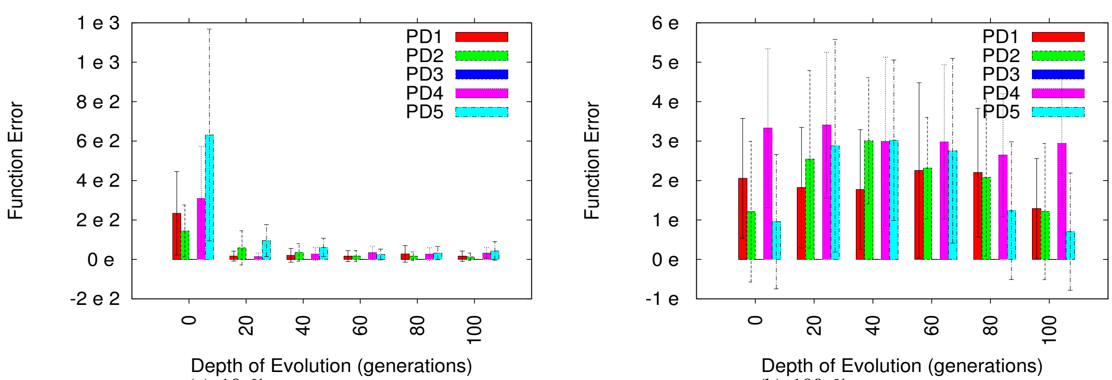
(a) 10 % convergence
(b) 100 % convergence

Fig. 3. Depth of evolution vs error for Shifted Sphere function [*f2*] at different stages of convergence.



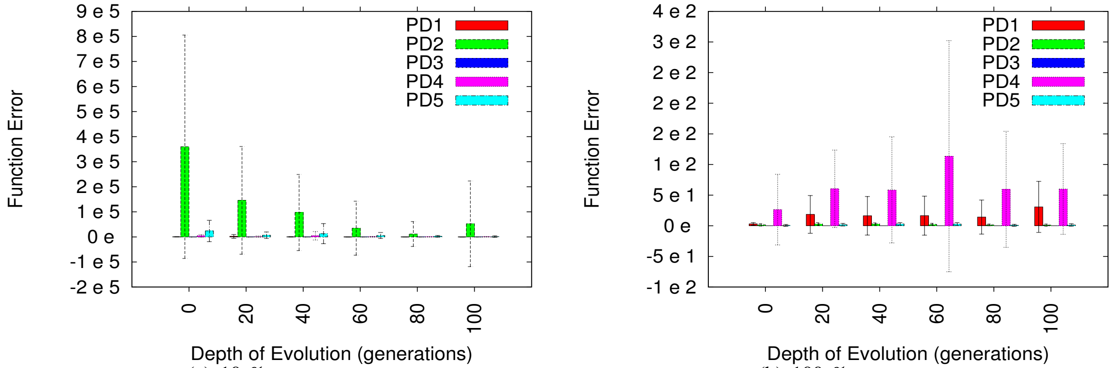
(a) 10 % convergence
(b) 100 % convergence

Fig. 4. Depth of evolution vs error for Schwefels Problem 1.2 function [*f3*] at different stages of convergence.



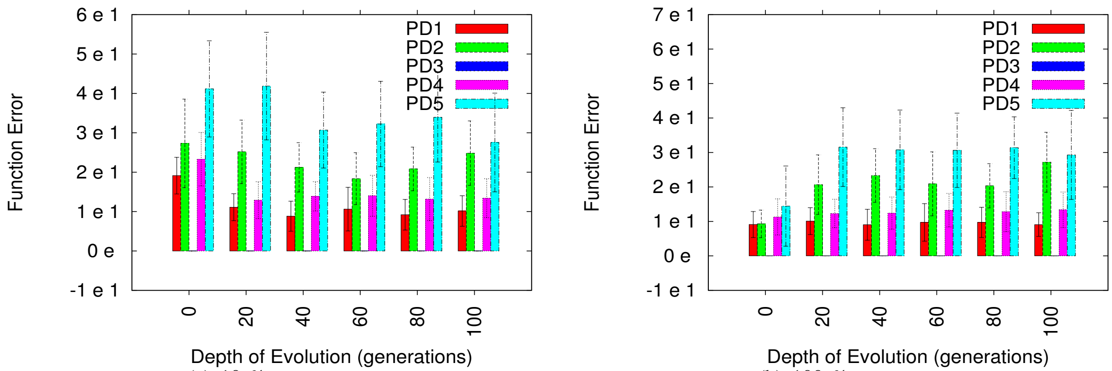
(a) 10 % convergence
(b) 100 % convergence

Fig. 5. Depth of evolution vs error for Rosenbrock function [*f4*] at different stages of convergence.



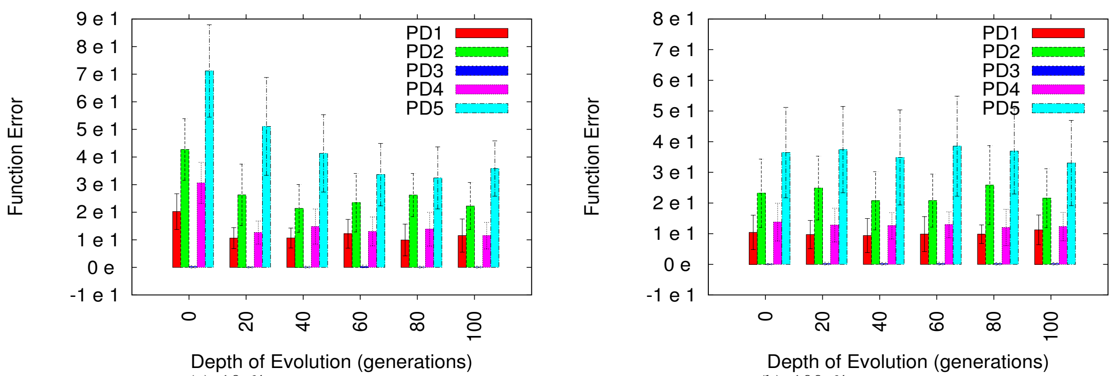
(a) 10 % convergence (b) 100 % convergence

Fig. 6. Depth of evolution vs error for Shifted Rosenbrock function [*f5*] at different stages of convergence.



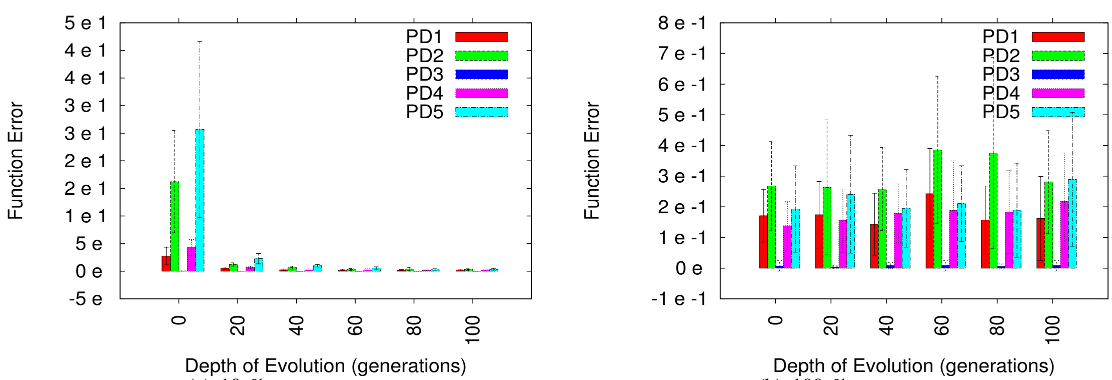
(a) 10 % convergence (b) 100 % convergence

Fig. 7. Depth of evolution vs error for Rastrigin function [*f6*] at different stages of convergence.



(a) 10 % convergence (b) 100 % convergence

Fig. 8. Depth of evolution vs error for Shifted Rastrigin function [*f7*] at different stages of convergence.



(a) 10 % convergence (b) 100 % convergence

Fig. 9. Depth of evolution vs error for Griewank function [*f8*] at different stages of convergence.

produced a larger error at a smaller depth of evolution in the initial stages of optimisation. Towards the end of evolution or optimisation, the depth of evolution did not play any significant role in the identifying the final error(minimum function error).

Fully separable and multi-modal functions, on the contrary, were unaffected by the depth of evolution. The error at all the depth of evolution investigated was almost the same. These type of problems had the same behavior throughout the optimisation process. It was also seen that towards the end of evolution, the depth does not play a major part for most problem decomposition strategies in all functions.

The results, in general, show that different problem decomposition methods have strengths and weakness at different stages of evolution. In some cases, some problem decomposition methods perform poorly and then become helpful at the end of evolution. This implies that the nature of the fitness landscape in terms of separability changes at different stages of evolution. Moreover, it is also seen that different problem decomposition strategies are suitable or have strengths for the different nature of the problems, for instance, *PD2* is best for the beginning of evolution for Figure 4 but poor in Figure 5.

There is future scope at adapting the depth of evolution - as the fitness error stagnates when compared to past behavior of the sub-population, it may be important to either increase the depth of evolution or reduce it - this is a matter of investigation. In some cases, the diversity can increase by the reintroduction of genetic material and the size of the sub-population can also be increased. Moreover, the behavior of cooperative coevolution as the size of the problem in terms of dimension can also give important observations for developing robust heuristics for improving cooperative coevolution.

VI. Conclusion

This paper has investigated effects of subcomponent optimisation in cooperative coevolution for different types of problems in terms of separability and multi-modality. The depth of evolution in subcomponent optimisation was analysed for both fully separable and non-separable problems as well as those that were uni-modal and multi-modal nature. This analysis can be very helpful in developing heuristic algorithms that adapt the intensity or depth of evolution for different subcomponents during different stages of the evolutionary process.

There is a significant performance issue for different problem decompositions strategies during the beginning and end of evolution which implies that the nature of the problem changes in terms of separability. We need to evaluate further if this has any relationship in terms of multi-modality which needs to be defined, assessed and evaluated during the course of evolution.

In future work, the measure of diversity in the respective sub-population at different depth of evolution is also important. We can define diversity that is specific to cooperative coevolution and then observe the behavior at different stages of evolution.

References

[1] K. Deb, *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, 2001, vol. 16.

[2] D. E. Clark and D. R. Westhead, "Evolutionary algorithms in computer-aided molecular design," *Journal of Computer-Aided Molecular Design*, vol. 10, no. 4, pp. 337–358, 1996. [Online]. Available: http://dx.doi.org/10.1007/BF00124503

[3] R. E. Bellman, *Adaptive control processes: a guided tour*. Princeton University Press, 1961.

[4] I. Zelinka, "A survey on evolutionary algorithms dynamics and its complexity mutual relations, past, present and future," *Swarm and Evolutionary Computation*, vol. 25, pp. 2 – 14, 2015, sI: {RAMONA}. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210650215000437

[5] M. Lozano, D. Molina, and F. Herrera, "Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems," *Soft Computing*, vol. 15, no. 11, pp. 2085–2087, 2011.

[6] M. Potter and K. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature PPSN III*, ser. Lecture Notes in Computer Science, Y. Davidor, H.-P. Schwefel, and R. Mnner, Eds. Springer Berlin Heidelberg, 1994, vol. 866, pp. 249–257.

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

[8] 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

[9] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *IEEE Congress on Evolutionary Computation*, 2008, pp. 1663–1670.

[10] 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*, vol. 26, pp. 3123–3136, 2015.

[11] R. Chandra and M. Zhang, "Cooperative coevolution of Elman recurrent neural networks for chaotic time series prediction," *Neurocomputing*, vol. 186, pp. 116 – 123, 2012.

[12] 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.

[13] K. Tang, X. Li, N. Suganthan, P, Z. Yang, and T. Weise, "Benchmark functions for the cec'2010 special session

and competition on large scale global optimization," Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep., 2009. [Online]. Available: http://nical.ustc.edu.cn/cec10ss.php

[14] 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.

[15] M. N. Omidvar and X. Li, "Cooperative coevolutionary algorithms for large scale optimisation," Technical report, The Royal Melbourne Institute of Technology (RMIT), Tech. Rep.

[16] K. Tang, X. Yao, P. N. Suganthan, C. MacNish, Y. P. Chen, C. M. 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, http://nical.ustc.edu.cn/cec08ss.php.

[17] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization," Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep., 2009, http://nical.ustc.edu.cn/cec10ss.php.

[18] X. Li, K. Tang, M. N. Omidvar, Z. Yang, and K. Qin, "Benchmark functions for the CEC'2013 special session and competition on large-scale global optimization," RMIT University, Melbourne, Australia, Tech. Rep., 2013, http://goanna.cs.rmit.edu.au/ xiaodong/cec13-lsgo.

[19] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary computation*, vol. 1, no. 1, pp. 1–23, 1993.

[20] M. 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.

[21] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 2. IEEE, 2004, pp. 1980–1987.

[22] K. Tang, X. Yao, P. Suganthan, C. MacNish, Y. Chen, C. Chen, and Z. Yang, "Benchmark functions for the cec2008 special session and competition on large scale global optimization. nature inspired computation and applications laboratory, ustc," *Applicat. Lab., Univ. Sci. Technol. China, Hefei, China, Tech, rep*, 2007.

[23] R. Chandra and K. Bali, "Competitive two-island cooperative coevolution for real parameter global optimisation."

[24] R. Chandra, "Competition and collaboration in cooperative coevolution of elman recurrent neural networks for time-series prediction," 2015.

[25] 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.

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

[27] D. Ortiz-Boyer, C. Hervás-Martínez, and N. García-Pedrajas, "Cixl2: A crossover operator for evolutionary algorithms based on population features." *J. Artif. Intell. Res.(JAIR)*, vol. 24, pp. 1–48, 2005.

[28] S. Kok and C. Sandrock, "Locating and characterizing the stationary points of the extended rosenbrock function," *Evolutionary Computation*, vol. 17, no. 3, pp. 437–453, 2009.

[29] A. Griewank and P. L. Toint, "Partitioned variable metric updates for large structured optimization problems," *Numerische Mathematik*, vol. 39, no. 1, pp. 119–137, 1982.

[30] B. Colson and P. L. Toint, "Optimizing partially separable functions without derivatives," *Optimization methods and software*, vol. 20, no. 4-5, pp. 493–508, 2005.

[31] K. Tang, X. Yao, P. Suganthan, C. MacNish, Y. Chen, C. Chen, and Z. Yang, "Benchmark functions for the cec2008 special session and competition on large scale global optimization. nature inspired computation and applications laboratory, ustc," *Applicat. Lab., Univ. Sci. Technol. China, Hefei, China, Tech, rep*, 2007.

[32] 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.

[33] 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," *Last accessed: July*, 2010.

[34] "Smart bilo: Computational intelligence framework," accessed: 02-12-2015. [Online]. Available: http://smartbilo.aicrg.softwarefoundationfiji.org