On the Selection of Decomposition Methods for Large Scale Fully Non-separable Problems

Yuan Sun
Department of Mechnical Engineering
The University of Melbourne
Parkville, Australia
yuans2@student.unimelb.edu.au

Michael Kirley
Department of Computing and Information Systems
The University of Melbourne
Parkville, Australia
mkirley@unimelb.edu.au

Saman K. Halgamuge
Department of Mechnical Engineering
The University of Melbourne
Parkville, Australia
saman@unimelb.edu.au

ABSTRACT

Cooperative co-evolution is a framework that can be used to effectively solve large scale optimization problems. This approach employs a divide and conquer strategy, which decomposes the problem into sub-components that are optimized separately. However, solution quality relies heavily on the decomposition method used. In recent years, a number of decomposition methods have been proposed, which raises another research question: Which decomposition method is best for a given large scale optimization problem? In this paper, we focus on the selection of the best decomposition method for large scale fully non-separable problems. Four decomposition methods are compared on a suite of benchmark functions. We observe that the random grouping method obtains the best solution quality on the benchmark large scale fully non-separable problems.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization

Keywords

Algorithm selection, large scale global optimization, cooperative co-evolution, problem decomposition.

1. INTRODUCTION

Evolutionary algorithms (EAs) are meta-heuristics that can be used to solve a wide range of optimization problems. However, when the problem has a large number of decision variables – *large scale global optimization* – it becomes difficult for an EA to find the optimal solution [1, 2].

Cooperative co-evolution (CC) [3] has been used with some successes when tackling large scale global optimization (eg. [4]). In CC, the optimization problem is divided into sub-components that are evolved independently. The final solution is a concatenation of representatives from each of the sub-components.

In recent years, there is an increased interest in solving large scale problems. When using the CC framework, it has been shown that the overall performance is correlated with the decomposition method used [5, 4]. This raises the following research question: Given a large scale optimization problem, which decomposition method is the best?

In this paper, we focus on the selection of the best decomposition method for large scale fully non-separable problems. Many real-world and benchmark problems are large scale fully non-separable. Four decomposition methods are investigated: random grouping (G) [6], delta grouping (D) [7], differential grouping (DG) [4], extended differential grouping (XDG) [8]. A detailed description of the four decomposition methods will be given in Section 3. The four decomposition methods are embedded in a CC framework to solve a suite of benchmark large scale fully non-separable problems. We observe that the G method achieves the best or comparable solution quality on all of the benchmark functions.

The remainder of this paper is organized as follows. Section 2 defines the large scale fully non-separable problems. Section 3 describes the four decomposition methods in detail. Section 4 sets up the experiments and analyses the experimental results. Section 5 concludes the paper.

2. FULLY NON-SEPARABLE PROBLEMS

In this section, we describe the definition of large scale fully non-separable problems.

We start by investigating the form of decision variable interaction in large scale problems. We suggest that there are two distinct types of variable interactions as shown in Figure 1. In Type I interactions, the variables interact directly eg. x_1 and x_2 (or x_2 and x_3) interact directly. In Type II interactions, the variables have a form of indirect interaction eg. x_1 and x_3 are linked by x_2. We call the former *direct interaction* and the latter *indirect interaction*. The formal definition of interacting types is listed below:

DEFINITION 1. *In an objective function $f(\vec{X})$, decision variables x_i and x_j interact directly if $\exists$ a candidate solution $\vec{x}_*$, such that*

$$\frac{\partial f}{\partial x_i \partial x_j}\Big|_{\vec{x}_*} \neq 0, \tag{1}$$

denoted by $x_i \leftrightarrow x_j$. Decision variables x_i and x_j interact

GECCO '15, July 11 - 15, 2015, Madrid, Spain

DOI: http://dx.doi.org/10.1145/2739482.2768483

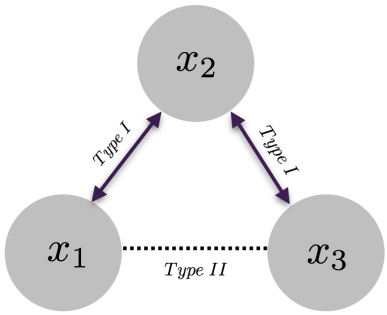


Figure 1: Two types of decision variable interaction. Type I: two variables interact directly with each other. Type II: two variables interact indirectly, that is they are linked via a third variable.

indirectly if for all candidate solutions,

$$\frac{\partial f}{\partial x_i \partial x_j} = 0, \tag{2}$$

and $\exists$ *a set of decision variables* $\{x_{k1}, \ldots, x_{kt}\} \subset \vec{X}$*, such that* $x_i \leftrightarrow x_{k1} \leftrightarrow \ldots \leftrightarrow x_{kt} \leftrightarrow x_j$*. Decision variables* x_i *and* x_j *are independent with each other if for all candidate solutions, (2) holds and* $\nexists$ *a set of decision variables* $\{x_{k1}, \ldots, x_{kt}\} \subset \vec{X}$*, such that* $x_i \leftrightarrow x_{k1} \leftrightarrow \ldots \leftrightarrow x_{kt} \leftrightarrow x_j$*.*

Consider the following example to further explain this definition:

EXAMPLE 1. *In the objective function:* $f(\vec{X}) = (x_1 - x_2)^2 + (x_2 - x_3)^2 + x_4^2$*,* $\vec{X} \in [-1, 1]^4$*,* x_1 *and* x_2 *interact directly with each other (Type I),* x_1 *and* x_3 *interact indirectly with each other (Type II),* x_1 *and* x_4 *are independent.*

The large scale fully non-separable problem is defined as follows:

DEFINITION 2. *In a large scale optimization problem, if all of the decision variables interact (directly or indirectly), it is large scale fully non-separable problem.*

3. DECOMPOSITION METHODS

In this section, we describe the four decomposition methods used within the CC framework in detail.

The G method [6] randomly assigns decision variables into sub-components. The number of sub-components is predetermined. In the beginning of each cycle, decision variable allocations are exchanged to increase the probability of assigning interacting decision variables into the same subcomponent. However, it has been shown that when the number of interacting variables is greater than two, it is unlikely to put all of them into the same sub-component [9].

The D method [7] assigns the decision variables into subcomponents in the evolutionary process. The number of sub-components is predetermined. It calculates the delta value[7] of each decision variable and places the decision variables with close delta values into the same sub-component.

The DG method [4] automatically assigns the interacting decision variables into the same sub-component. When changes in the fitness value caused by adding a perturbation to x_i varies for different values of x_j, then x_i and x_j are regarded as interacting decision variables. However the DG method fails to capture indirect interaction [8].

The XDG method [8] can capture both direct and indirect interaction between decision variables. XDG uses the same technique to identify direct interaction with DG. In XDG, after allocating decision variables that interact directly to nominated sub-components, "overlaps" between subcomponents are identified. That is, when an overlap is observed, the sub-components that contain the same decision variables are merged. This searching–merging technique is employed to capture indirect interactions between decision variables.

4. EXPERIMENTS

In this section, numerical experiments are conducted to investigate the following research questions:

Q1. What are the decomposition results obtained by the G/D/DG/XDG methods?

Q2. Which decomposition method is the best method to use when solving large scale fully non-separable problems when incorporated with a CC framework?

4.1 Methodology

To investigate the research questions, the 8 large scale fully non-separable benchmark functions are used (See Table 1). Function f_1 and f_2 are from the CEC'2010 special session on large scale global optimization [10]. Function f_3 to f_8 are widely used benchmark functions. The original functions of f_4 to f_8 are fully separable. Rotation operator is employed to generate interaction between decision variables. For the 8 benchmark functions, the dimensionality $d = 1000$; the domain is $[-100, 100]^d$; the optimal solution is $f(\vec{0}) = 0$.

To investigate Q1, the G/D/DG/XDG methods are selected to decompose the benchmark functions. The parameter value ϵ was set to 10^{-3}. The sub-components formed by G/D/DG/XDG on each benchmark function are recorded. In addition, the number of function evaluations (FE) used to decompose the problem is recorded.

To investigate Q2, Differential Evolution Cooperative Coevolution (DECC) is selected as the CC framework. DECC uses SaNSDE [11] to optimize each subcomponent. The population size is set to 50. The maximal number of FE is set to 3×10^6, divided between the decomposition phase and the evolutionary optimization phase. The G/D/DG/XDG methods are incorporated into the DECC framework to solve the benchmark functions. For DECC-G/D/DG/XDG, 25 independent runs are conducted for each benchmark function. The mean of the best solutions found in the fixed number of FE are recorded to evaluate the performance of the algorithms. The two-sided Wilcoxon test with the confidence interval of 95% is used to determine the significantly best performance from the four algorithms in a pairwise fashion.

4.2 Decomposition Comparison

In this section, we present the decomposition results of the G/D/DG/XDG methods on the 8 benchmark functions (See Table 2). In Table 2, "Group" represents formed groups, which is $s \times n$, where s is the size of groups, and n is the number of groups. "FE" represents the number of FE used in the decomposition phase.

The G method randomly divides the 1000 interacting decision variables into 10 groups, each with 100 decision vari-

Table 1: Benchmark functions

Func_ID	Name	Equation	Domain	Optimum
f_1	Shifted Schaffer	$f(x) = \sum_{i=1}^{d} \sum_{j=1}^{i} x_j^2$	$[-100, 100]^d$	$f(\vec{0} = 0)$
f_2	Shifted Rosenbrock	$f(x) = \sum_{i=1}^{d-1} 100(x_i^2 - x_{i+1})^2 + \sum_{i=1}^{d-1} (x_i - 1)^2$	$[-100, 100]^d$	$f(\vec{0} = 0)$
f_3	Griewank	$f(x) = 1 + \frac{1}{4000} \sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos \frac{x_i}{\sqrt{i}}$	$[-100, 100]^d$	$f(\vec{0} = 0)$
f_4	Sharp Ridge	$f(x) = x_1^2 + 100\sqrt{\sum_{i=2}^{d} x_i^2}$	$[-100, 100]^d$	$f(\vec{0} = 0)$
f_5	Rotated Elliptic	$f(x) = \sum_{i=1}^{d} 10^{6\frac{i-1}{d-1}} x_i^2$	$[-100, 100]^d$	$f(\vec{0} = 0)$
f_6	Rotated Rastrigin	$f(x) = 10d - 10 \sum_{i=1}^{d} \cos(2\pi x_i) + \sum_{i=1}^{d} x_i^2$	$[-100, 100]^d$	$f(\vec{0} = 0)$
f_7	Rotated Alpine	$f(x) = \sum_{i=1}^{d} \lvert x_i \sin x_i + 0.1 x_i \rvert$	$[-100, 100]^d$	$f(\vec{0} = 0)$
f_8	Rotated Ackley	$f(x) = 20 - 20 \exp -0.2\sqrt{\frac{\sum_{i=1}^{d} x_i^2}{d}} - \exp \frac{\sum_{i=1}^{d} 2\pi x_i}{d} + e$	$[-100, 100]^d$	$f(\vec{0} = 0)$

Table 2: The decomposition results of the G/D/DG/XDG methods on benchmark functions.

Func		DECC-G	DECC-D	DECC-DG	DECC-XDG
f_1	Group	100×10	100×10	1000×1	1000×1
	FE	0	0	2000	3998
f_2	Group	100×10	100×10	2×500	1000×1
	FE	0	0	501000	1001000
f_3	Group	100×10	100×10	1000×1	1000×1
	FE	0	0	1001000	1001000
f_4	Group	100×10	100×10	1000×1	1000×1
	FE	0	0	1001000	1001000
f_5	Group	100×10	100×10	1000×1	1000×1
	FE	0	0	2000	3998
f_6	Group	100×10	100×10	1000×1	1000×1
	FE	0	0	2000	3998
f_7	Group	100×10	100×10	1000×1	1000×1
	FE	0	0	2000	3998
f_8	Group	100×10	100×10	1000×1	1000×1
	FE	0	0	2034	4030

ables, at the beginning of each cycle. It does not need any FE in the decomposition phase.

The D method decomposes the optimization problem in the evolution process. It places 100 decision variables into one group according to the delta values. Therefore, it also divides the 1000 decision variables into 10 groups, each with 100 decision variables. It does not need any extra FE in the decomposition phase.

The DG method identifies the interaction between decision variables before the evolutionary process starts. It places all directly interacting decision variables into one group. As shown in Table 2, on f_1, f_5, f_6, f_7 and f_8, DG successfully identifies all the 1000 decision variables as interacting and places them into a large group. On f_2, DG forms 500 groups, each with 2 decision variables. The reason is that f_2 is a Rosenbrock function, which contains indirect interaction. DG can not identify indirect interaction [8]. On f_3 and f_4, the number of FE used in the decomposition phase is very large (1001000). The reason is that DG unexpectedly identifies all the 1000 decision variables as separable. However, DG places all separable variables into one group.

Table 3: The optimization results of the DECC-G/D/DG/XDG algorithms on the benchmark functions. Better performances are highlighted in bold.

Func	DECC-G	DECC-D	DECC-DG	DECC-XDG
f_1	**2.39e-04**	1.48e+06	1.01e+05	8.20e+04
f_2	**3.98e-06**	2.81e+03	9.51e+02	5.40e+07
f_3	**3.10e-15**	**2.77e-15**	1.62e+00	4.77e-01
f_4	6.44e-11	**1.05e-11**	3.71e+03	2.54e+03
f_5	**2.21e-02**	2.51e+08	5.59e+07	5.35e+07
f_6	**3.35e-14**	1.63e+04	1.03e+05	1.14e+05
f_7	**3.16e-03**	1.47e+04	6.58e+03	6.44e+03
f_8	**3.16e-13**	2.15e+01	2.15e+01	2.15e+01

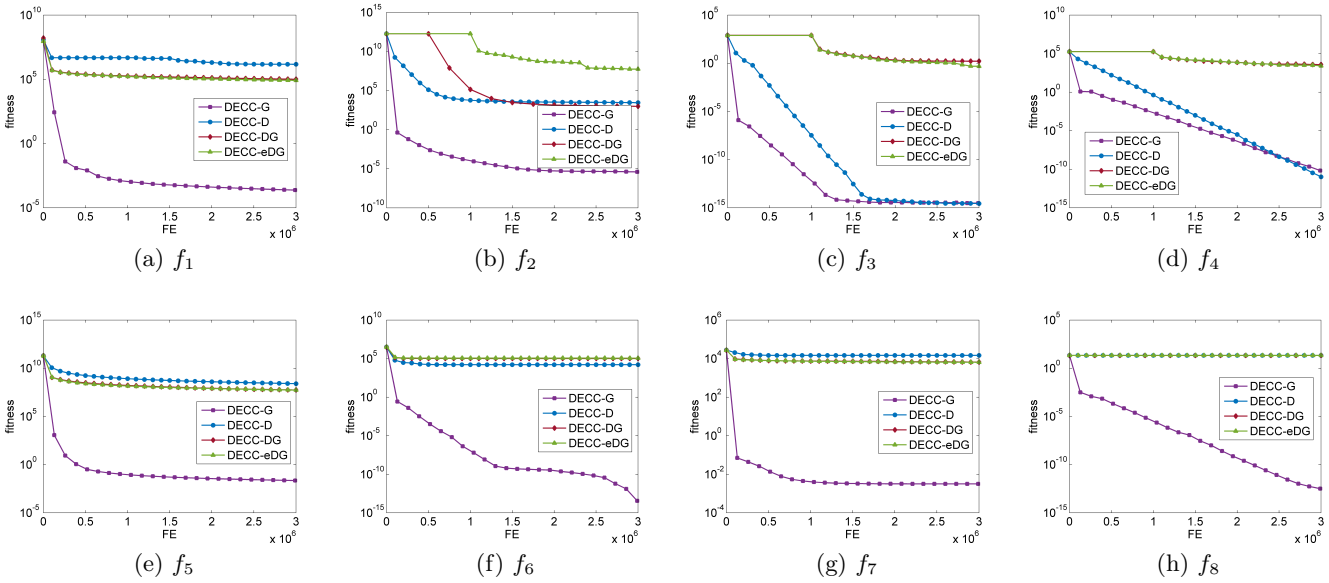
Therefore, DG also forms a large group with 1000 decision variables on f_3 and f_4.

The XDG method obtains the same decomposition with DG on the benchmark functions except f_2. On f_2, XDG successfully identifies all decision variables as interacting and places them in a large group. Note that the number of FE used by XDG is greater than DG. The extra FE is used to identify indirect interaction.

4.3 Optimization Comparison

In this section, we present the optimization results of the DECC-G/D/DG/XDG algorithms on the benchmark functions (See Table 3). In Table 3, the best results are highlighted in bold. The convergence plots of the four algorithms on the benchmark functions are shown in Figure 2.

As shown in Table 3 and Figure 2, The DECC-G algorithm achieves the best performances on 7 out of 8 benchmark functions. On function f_4, DECC-G obtains comparable result with DECC-D. The DECC-D algorithm achieves the best results on function f_3 and f_4. Note that both G and D divide the 1000 interacting decision variables into 10 groups, each with 100 decision variables. However, DG and XDG places all the 1000 interacting decision variables into one large group in most cases. It may indicate that when



(a) f_1 (b) f_2 (c) f_3 (d) f_4

(e) f_5 (f) f_6 (g) f_7 (h) f_8

Figure 2: The convergence plots of the DECC-G/D/DG/XDG algorithms on the benchmark functions.

the number of interacting decision variables is large, it is more efficient to divide them into several sub-components.

5. CONCLUSION

In this paper, we focus on the selection of the best decomposition method for large scale fully non-separable problems. The G/D/DG/XDG methods are embedded in the DECC framework to solve a suite of benchmark functions. The experimental results show that the DECC-G algorithm achieves the best or comparable results on all of the benchmark functions. We conclude that the G method should be selected for large scale fully non-separable problems.

The scope of this paper is limited. On the large scale fully separable or partially separable problems, it is not likely that the G method is the best decomposition method. The first step of future work is to select the best decomposition method for any large scale optimization problem. More decomposition methods should be included as the candidates.

The second step of future work is to select the best algorithm for a given large scale optimization problem. This paper is in the context of DECC framework. Other CC frameworks should be included in the algorithm space. In addition, other well-performed algorithms (not limited to CC) should also be included in the algorithm space.

6. REFERENCES

[1] T. Weise, R. Chiong, and K. Tang, "Evolutionary Optimization: Pitfalls and Booby Traps," *Journal of Computer Science and Technology*, vol. 27, pp. 907–936, Nov. 2012.

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

[3] M. Potter and K. D. Jong, "A cooperative coevolutionary approach to function optimization," *Parallel Problem Solving from Nature - PPSN III*, pp. 249 – 257, 1994.

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

[5] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in *Lecture Notes in Computer Science*, pp. 300–309, 2010.

[6] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences*, vol. 178, pp. 2985–2999, Aug. 2008.

[7] M. N. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping for large scale non-separable function optimization," in *IEEE Congress on Evolutionary Computation*, 2010.

[8] Y. Sun, M. Kirley, and S. K. Halgamuge, "Extended Differential Grouping for Large Scale Global Optimization with Direct and Indirect Variable Interactions," in *the conference on Genetic and evolutionary computation. ACM*, 2015.

[9] M. N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative co-evolution for large scale optimization through more frequent random grouping," in *IEEE Congress on Evolutionary Computation*, 2010.

[10] K. Tang, X. Yáo, and P. Suganthan, "Benchmark functions for the CEC'2010 special session and competition on large scale global optimization," *Rapport technique, USTC, Nature Inspired Computation and Applications Laboratory*, no. 1, pp. 1–23, 2010.

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