Large-Scale Evolutionary Optimization via Multi-Task Random Grouping

Phoenix Neale Williams, Ke Li and Geyong Min

***Abstract*—Evolutionary Algorithms (EA) are known to suffer from the *curse of dimensionality* resulting in poor performances when handling large-scale problems. Cooperative coevolution aims to overcome these issues in a divide and conquer approach by decomposing the original problem into several lower-dimensional sub-problems. For each sub-problem a chosen EA is applied for a defined number of function evaluations. This is repeated in a *round robin* like fashion until a terminating condition is met. A recently proposed area in the evolutionary computation field is the evolutionary multi-task optimization (EMTO) framework. By jointly optimising several tasks, EMTO aims to exploit beneficial information across multiple tasks to improve the performance compared to optimising each task in isolation. In this paper, we consider a large-scale problem as a multi-task optimization problem by considering each sub-problem as an independent task. Applying an EMTO algorithm, knowledge transfer across sub-problems is carried out explicitly to improve the optimization of each sub-problem. We evaluate the effectiveness of our proposed algorithms empirically on a suite of separable and non-separable benchmark problems of varying dimensions.**

I. Introduction

The large-scale black-box optimization problems considered in this paper is defined as follows:

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x}) \\ &\text{subject to} \quad \mathbf{x} \in \Omega \ , \end{aligned} \tag{1}$$

where $\Omega = [x_i^L, x_i^U]^d \subseteq \mathbb{R}^d$ defines the search (or decision variable) space and d is reasonably large, in our case $d \geq 100$. Evolutionary algorithms (EA) are a family of derivative-free optimization algorithms based on the evolution theory of Wallace and Darwin [1]–[3] that have been successfully applied to low and moderate dimensional problems ($d \leq 100$). In literature the performance of evolutionary algorithms have shown to deteriorate when the search space grows [4], termed the *curse of dimensionality*.

To overcome these issues Cooperative co-evolution (CC) [5] based algorithms follow a divide and conquer approach. By decomposing the original large-scale problem into several lower-dimensional sub-problems a CC algorithm applies and EA to each sub-problem for a defined number of function evaluations. In literature the optimization of all sub-problems is termed a *cycle*. A CC based algorithm can carry out several *cycles* during the optimization of a large-scale problem. By optimizing each sub-problem in turn the original large-scale problem is solved. The evaluation of a sub-problem solution is carried out by inserting its variables into the best solution currently discovered in the original high-dimensional search space, the updated high-dimensional solution is then evaluated on the objective function. Further research employing the CC approach has primarily consisted of developing improved decomposition strategies to improve the accuracy of grouping correlated variables together into a single sub-problem.

Evolutionary multi-task optimization (EMTO) is a recently proposed research area [6] that studies the use of evolutionary algorithms [7] to jointly solve multiple optimization problems simultaneously by exchanging knowledge between tasks. Through the exploitation of knowledge from distinct tasks an EMTO algorithm aims to improve the optimization performance compared to optimizing each task independently. To carry out knowledge transfer, EMTO algorithms apply genetic operators on solutions of different tasks. Work in literature make use of either implicit of explicit knowledge transfer. By maintaining a single population of solution evaluated on different tasks, implicit methods make use of *assortative mating* to generate offspring's. Assortative mating applies a crossover operator on solutions of different tasks at a probability of rmp or only mutates solutions if the condition of knowledge transfer is not met. Conversely, explicit methods maintain independent populations for each task inserting solutions into the evolution process of a different tasks population for knowledge transfer at a rate of rmp. In both approaches the rate of knowledge transfer is fully described by the rmp parameter that represents the probability of knowledge transfer between tasks occurring.

Work using the CC approach for large-scale problems aim to optimize each sub-problem independently in an iterative process updating the best found solution in the original space after the optimization of a sub-problem. Considering each sub-problem as an independent task, we aim to jointly optimize each sub-problem simultaneously through a multi-task optimization approach. By first generating several sub-problems through the random grouping strategy, we jointly optimize each sub-problem simultaneously using a multi-task evolutionary algorithm inspired by the DEMTO algorithm of Zheng et al [8], termed MTEA-RG. Using a suite of benchmark functions we study the impact the rmp parameter has on the performance of the MTEA-RG algorithm. Using the same suite of benchmark functions we show the improved performance the MTEA-RG algorithm offers compared to other peer algorithms.

The rest of this paper is organised as follows. Section II briefly overviews current work in both large-scale evolution-

This work was supported by UKRI Future Leaders Fellowship (MR/S017062/1) and EPSRC Doctoral Training Partnership (2404317).

P.Williams and K. Li are with the Department of Computer Science, University of Exeter, Exeter, EX4 4QF, UK {pw384,k.li,g.min}@exeter.ac.uk

ary computation using the CC approach and evolutionary multi-task optimization giving details of their implementations. The proposed MTEA-RG algorithm is outlined in Section III with Section IV investigating the effectiveness of the algorithm by conducting a parameter study and comparison with other peer algorithm. Finally, Section V concludes the paper with a summary and directions of future research.

II. Preliminaries

A. Literature Review

The first CC algorithm proposed by Van den Bergh and Engelbrecht [9] split the original $D-$dimensional problem into $D, 1-$dimensional sub-problems. In the proposed work, the authors showed the limitation of their approach in solving non-separable problem, namely the Rosenbrock function. Addressing this, Yang et al. [4] proposed a random grouping strategy that generates S, d_g-dimensional sub-problems, where each variable x_i has an equal probability of being grouped together. To increase the probability of discovering the true decomposition, the authors randomly decompose the original problem for each cycle. A drawback of this method was the algorithms sensitivity to the S and d_g parameters values. Enforcing that the original dimension $D = S \times d_g$, the random grouping strategy is fully described by the S value. To improve the decision making of the S parameter, Yang et al. [10] proposed an adaptive random-grouping strategy that selects the S value from a list of possible values before each cycle. Providing each S value with an initial equal probability of being selected, the algorithm updates the probability of each S based on its performance during the optimization process using a reward strategy. Whereas the random grouping strategy computed each sub-problem in a stochastic manner, the delta-groping strategy of Omidvar et al. [11] estimates an improvement interval for each dimension which is updated throughout the optimization process. Prior to each cycle, all dimensions are sorted based on their improvement interval grouping variables with similar improvement intervals grouped together. Another strategy developed by Omidvar et al. termed differential grouping, [12] grouped together variables based on the impact of their variance on the objective function. Sun et al. [13] proposed an extension to the differential grouping strategy by categorizing the interaction of variables as direct, indirect or separable. Ling et al [14] applied the DG condition of interacting variables to form a graph using graph-theory to decompose the problem. The core drawback of grouping methods such as differential grouping is the computational cost of its use. Applying DG to a fully separable $D-$dimensional problem requires $2 \times D$ function evaluations. Addressing this issue Omidvar et al. [15] extended their work to improve the computational expense of differential grouping whilst also allowing for sub-problems to overlap.

Evolutionary multi-task optimization (EMTO) [6] applies an evolutionary algorithm to jointly optimize multiple tasks simultaneously. By exploiting information between the various tasks, EMTO aims to improve optimization performance compared to solving each task independently. Information between tasks are exchanged by applying genetic operators on solutions of different tasks with a probability of rmp. Algorithms within the EMTO framework can be categorized based on their use of either implicit of explicit genetic knowledge transfer. The original EMTO work of Gupta et al. [6] employs implicit knowledge transfer which maintains a single population of solutions evaluated on different tasks. Explicit knowledge transfer takes multi-population approach and maintains a single population for each task. To carry out knowledge transfer, solutions are inserted into the evolution process of populations of the other tasks. In both approaches, the rate of knowledge transfer between different tasks is controlled by the rmp parameter, therefore the performance of an EMTO algorithm is highly sensitive to the rmp parameter value. An overly low rmp value over restricts the amount of knowledge transferred between task, whereas a large rmp value may result in an overly large amount of knowledge transfer. In both cases the result is the premature convergence of the EMTO algorithm. The latter condition is commonly known as *negative transfer*. Addressing the issue of *negative transfer* Wen et al. [16] proposed an adaptive EMTO method that updates the rmp value throughout the optimization process. By tracking the survival rate of solutions produced from different tasked solutions the rmp parameter is updated. When the survival rate drops below a defined threshold the rmp value is set to 0. For an EMTO algorithm handling more than 2 tasks, it is unlikely a single rmp is able to fully describe the different relationships between tasks. Addressing this Bali et al. [17] proposed a matrix RMP consisting of rmp values between tasks. Assuming that the distribution of offspring's should resemble the distribution of the parent population, the authors model each parent population as a Gaussian mixture model incorporating the RMP matrix. To estimate the RMP matrix the authors maximize the log-likelihood function. Beyond the fitting of rmp value, another direction of research within the EMTO framework is the defining of space for all tasks. Original works involve the normalization of the search space of each task to the range [0, 1] and projecting a solution into the search space of a task for evaluation. Bali et al. [18] proposed a method of projecting solutions into the search space of a tasks before applying genetic operators, removing the requirement of a unified search space for all tasks. Other works have applied different evolutionary algorithms to the multi-task setting such as differential evolution [8], [19] and particle swarm optimization [19].

As used within the CC approach, decomposing a high-dimensional problem results in several lower-dimensional problems that require solving. With the addition of a changing best current solution, this environment can be described as a special case multi-task optimization problem. Works in current literature handle this by optimizing each sub-

Algorithm 1: Cooperative Coevolution Approach

Input: Number of groups n, problem dimension d, population size $popsize$, function to be optimized $func$, evolutionary optimizer $optimizer$

Output: Estimated Optimum $best$

$groups \leftarrow grouping(d, n)$ // grouping stage
$pop \leftarrow rand(popsize, n)$ // Initial random population
$(best, best\_val) \leftarrow min(func(pop))$
for $i = 1$ *to cycles* **do**
  for $j = 1$ *to size(groups)* **do**
    $indicies \leftarrow groups[j]$
    $subpop \leftarrow pop[:, indices]$
    $subpop \leftarrow optimizer(best, subpop, FE, func)$
    $pop[:, indicies] \leftarrow subpop$
    $(best, best\_val) \leftarrow min(func(pop))$
return best

problem independently, updating the best solution after the optimization of each sub-problem. In comparison we aim to jointly optimize each sub-problem by employing an EMTO algorithm to this multi sub-problem setting. Applying this interpretation, we transform the transfer of knowledge between tasks to the knowledge between dimensions.

B. Cooperative Coevolution

An algorithm employing cooperative co-evolution begins by decomposing the original high-dimensional problem into several sub-problems using a chosen variable grouping strategy. In each *cycle* each sub-problem is optimized in terms of the best found solution in the original space by applying a chosen Evolutionary Algorithm for a defined number of function evaluations. An algorithm employing a random grouping strategy may re-generate the set of sub-problems prior to each cycle to improve the probability of generating the true decomposition. Solutions in each sub-problem are evaluated by inserting its variables into the best currently found solution in the original space at locations specified by the grouping of the sub-problem. Once the high-dimensional solution has been updated with a solutions variable it is evaluated on the objective function. Each cycle concludes by setting the best current solution in the original space. An algorithm may carry out several cycles during the whole optimization process. Pseudo-code of a CC algorithm is described in Algorithm 1.

C. Evolutionary Multi-Task Optimization

An evolutionary multi-task algorithm with explicit genetic transfer (EMTOe) maintains and evolves a single population of solutions for each task. The pseudo-code of the algorithm is given in Algorithm 2. Considering the optimization of k task, an EMTOe algorithm begins by generating an initial population for each of the k tasks. Whilst the termination

Algorithm 2: EMTOe Algorithm

Input: Size of population N, probability of knowledge transfer rmp, set of tasks to be optimized T, number of tasks k, evolutionary optimizer EA, solution selection policy SE

Output: Estimated Optimum x_t^* for each task t

Initialize population pop_j with N individuals for task $T_j, j = 1, 2, ..., k$
Evaluate each individual in pop_j on task T_j
while *stopping condition is not satisfied* **do**
  for *task j=1 to k* **do**
    $rand \leftarrow random(0, 1)$
    if $rand < rmp$ **then**
      $M \leftarrow \{\}$
      for *task i = 1 to k* **do**
        if $i \neq j$ **then**
          $M \leftarrow M \cup \{SE(pop_i)\}$
      $pop_j \leftarrow EA(pop_j, M)$
    else
      $pop_j \leftarrow EA(pop_j)$

condition is not satisfied an EMTOe algorithm iteratively evolves each population using a defined evolutionary optimizer EA for single generations in a similar way to CC based algorithm. For each task, a random float $rand$ within the range $[0, 1)$ is generated. If the condition $rand < rmp$ is satisfied solutions of different tasks are selected through a selection policy SE and inserted into the optimization process of the evolutionary optimizer EA. This is continued until convergence or all computational resources have been used.

III. Proposed Method

In this section, we propose the large-scale evolutionary optimization via multi-task random grouping (MTEA-RG) algorithm. Given a set of sub-problems generated through the random grouping strategy of a large-scale problem, the MTEA-RG aims to jointly optimize each sub-problem simultaneously in a multi-task approach.

Given a real-valued function f, the MTEA-RG algorithm begins by decomposing the original high-dimensional problem using random grouping into S non-overlapping sub-problems. The MTEA-RG algorithm initializes by filling the population of each sub-problem with randomly generated solutions within the problem search space. The MTEA-RG algorithm then generates a random $D-$dimensional *component vector* CV within the problems search space. For each sub-problem p_i with dimensions d_i, a solution s_j^i is evaluated by inserting its variables into the *component vector* at locations specified by d_i. For each evaluation the *component vector* is updated to be the solution with the best objective function value. This process is followed

Algorithm 3: MTEA-RG Algorithm

Input: probability of knowledge transfer rmp, population size for each group $\mathrm{N_p}$, the number of groups S, the dimension of the problem d

Output: Estimated Optimum CV

```
G ← grouping(d, S)
Initialize population pop_j with N_p individuals for group G_j, j = 1, 2, .., S
Evaluate each individual in pop_j and set CV as the best solution in the original space
while termination condition not satisfied do
    for group j = 1 to S do
        Apply rand/1 mutation operator to pop_j and generate N_p mutant vectors
        if rand < rmp then
            Replace mutant vectors with random vectors selected from the best vectors of the other S − 1 groups
        Apply binomial crossover to the set of mutant vectors to generate N_p trial vectors
        Evaluate the set of trial vectors and set CV as the best solution in the original space
        Combine pop_j and set of trial vectors
        Set pop_j as the best N_p solutions from combined population
        Update the best solution found for group G_j
return CV
```

for the evaluation of every solution. In a round-robin like fashion, each population is evolved independently using the differential evolution [2] algorithm with a $rand/1$ mutation strategy and binomial crossover. For each generated mutant population the condition for knowledge transfer is checked. If the condition is satisfied, the mutant population is completely replaced by randomly selected solutions from a set of best performing solutions from the other $S-1$ populations. Offspring solutions are generated using the mutant population through binomial crossover with the parent population. Following the EMTO framework the rate of knowledge transfer in the MTEA-RG algorithm is fully controlled by the rmp parameter. Elitist selection is applied to the population of parent and offspring solutions generating a population for the next generation. The pseudo-code of the proposed MTEA-RG algorithm is shown in Algorithm 3

IV. Empirical Studies

We begin this section by conducting a parameter study, analysing the impact the rmp parameters value has on the proposed MTEA-RG algorithm. To evaluate the algorithms performance we carry out this study on a suite of non-separable and separable benchmark problems. The grouping size k, number of groups S and population size $\mathrm{N_p}$ used in these experiments are described in Table I.

TABLE I: The experimental set-up of the N, k and S values for each dimension

d	N_p	k	S
100	50	10	10
300	40	30	10
500	30	50	10
1000	20	100	10

A. Parameter study

For the conducted experiments the MTEA-RG algorithms make use of the *rand/1* mutation operators with parameters F and CR set to 0.6 and 0.9 respectively. For each benchmark problem we run the MTEA-RG algorithm for $300,000$ function evaluations.

From the convergence plots in figure 1 we see that raising the rmp value increases the frequency of the knowledge transfer across sub-problems. Increasing the rmp value beyond 0.0 we see a jump in performance in the MTEA-RG algorithm on all benchmark problems. This behaviour gives the intuition that different sub-problems hold information that can improve the optimization process of other sub-problems. As each sub-problem corresponds to a sub-set of variables of the original large-scale problem we describe such a situation as beneficial *cross-dimensional* information. The level of beneficial *cross-dimensional* information however may differ for various problems, this is evident from the convergence plots shown in figure 1.

We also see the negative impact of overly high frequencies of knowledge transfer. In combination with the elitist selection the algorithm experiences a premature convergence on several problems for larger rmp values. From these plots we see that rmp value can greatly impact the performance of the MTEA-RG algorithm. This opens the opportunity for further research.

The convergence plots in figure 1 reinforce the previous hypothesis of the sensitivity in the MTEA-RG algorithms performance caused by the rmp parameter. Similar to the *no free lunch theorem* a well performing rmp value is dependent on the optimization problem. Additionally it is likely rmp between various sub-problems would be better suited to this set-up. This leads to further questions that can be addressed within this new consideration of a large-scale optimization problem.

We provide a full table of experiment results in the supplementary material. The results show the positive impact knowledge transfer has on the optimization performance of a large-scale problem whilst also opening doors for further research in key areas. For comparison with peer algorithm we set $rmp = 0.3$ due to its frequency in achieving the best performance on the suite 56 benchmark experiments.

B. Comparison with peer-algorithms

We compare the performance of the proposed MTEA-RG algorithm for $rmp = 0.3$ with the EMT-RE [20] algorithm and a CC algorithm with differential grouping and a differential evolution sub-problem optimizer (CC-DG-DE). The MTEA-RG algorithm settings follow the set-up outlines

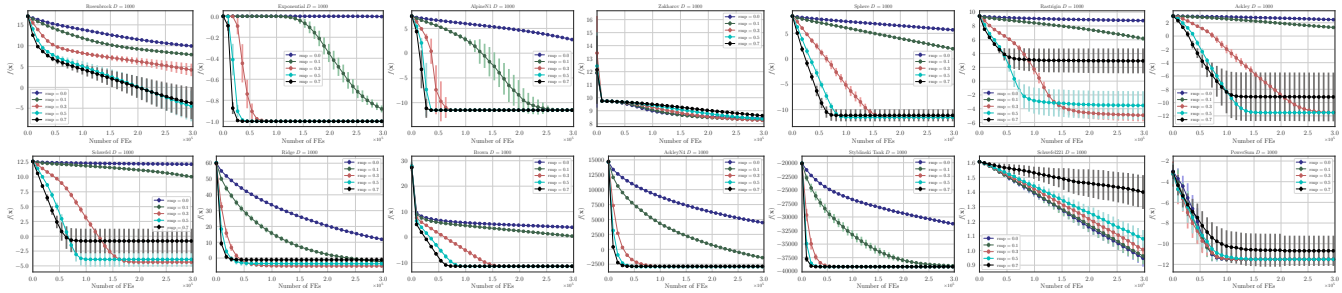

Fig. 1: MTEA-RG mean and standard deviation with differing rmp values on benchmark problems with $D = 1000$

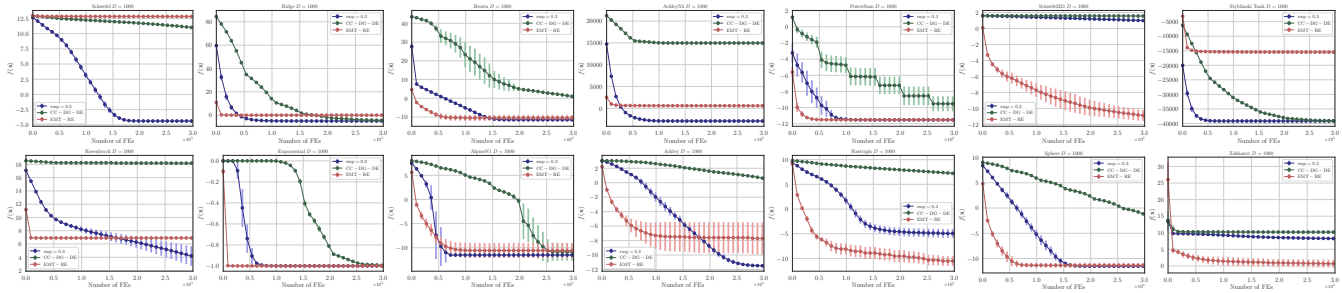

Fig. 2: Mean and standard deviation comparison with peer algorithms on benchmark problems with $D = 1000$

	d	MTEA-RG	EMT-RE	CC-DG-DE
Rosenbrock	100	**9.659E+1(4.57E-2)**	9.873E+1(4.81E-2)	1.746E+4(5.92E+3)
	300	**2.950E+2(3.27E+2)**	2.988E+2(7.10E-2)	1.243E+6(2.89E+5)
	500	**1.460E-2(4.94E+2)**	4.987E+2(1.43E-1)	1.684E+3(2.90E+2)
	1000	**3.238E+2(9.85E+2)**†	9.988E+2(8.55E-2)	8.351E+7(2.89E+6)
Ridge	100	**-5.000E+0(0.00E+0)**†	2.309E-2(2.76E-2)	-5.0E+0(9.54E-7)
	300	**-5.000E+0(4.77E-07)**†	1.185E-3(5.06E-3)	-5.000E+0(1.79E-6)
	500	**-5.000E+0(1.43E-6)**†	5.516E-4(2.45E-3)	-4.999E+0(9.33E-5)
	1000	**-5.000E+0(4.16E-6)**†	1.631E-4(8.54E-4)	-4.624E+0(1.63E-2)
Schwefel	100	**2.368E+3(1.30E+3)**†	2.721E+4(4.27E+3)	1.407E+4(1.07E+3)
	300	**3.818E-3(0.00E+0)**†	9.847E+4(2.55E+3)	4.938E+4(4.46E+3)
	500	**3.364E-4(4.66E-10)**	1.729E+5(5.71E+4)	2.197E+3(5.93E+2)
	1000	**1.273E-2(5.59E-9)**†	3.695+5(1.21E+4)	5.365E+4(2.64E+3)
Alpine N1	100	**0.000E+0(0.00E+0)**	1.764E-2(6.58E-2)	**0.000E+0(0.00E+0)**
	300	**0.000E+0(0.00E+0)**	1.848E-5(3.33E-4)	**0.000E+0(0.00E+0)**
	500	**0.000E+0(0.00E+0)**	9.158E-7(2.09E-5)	**0.000E+0(0.00E+0)**
	1000	**0.000E+0(0.00E+0)**	3.872E-6(1.67E-5)	**0.000E+0(0.00E+0)**
Ackley N4	100	**-2.803E+2(1.11E+1)**†	-1.014E+2(1.32E+1)	2.906E+2(1.31E-2)
	300	**-8.742E+2(1.63E-1)**	-2.600E+1(2.58E+1)	-8.742E+2(1.02E+0)
	500	**-1.458E+3(9.86E-3)**†	1.186E+2(3.42E+1)	3.547E+3(1.46E+3)
	1000	**-2.917E+3(1.67E-1)**†	5.820E+2(5.54E+1)	1.496E+4(2.03E+2)
Brown	100	**0.000E+0(0.00E+0)**	3.221E-5(8.33E-4)	**0.000E+0(0.00E+0)**
	300	**0.000E+0(0.00E+0)**†	2.050E-6(3.03E-5)	5.165E-13(4.22E-13)
	500	**8.716E-27(3.87E-26)**†	1.805E-7(2.07E-5)	7.584E-6(3.31E-5)
	1000	**2.295E-15(5.10E-15)**†	3.080E-7(4.37E-5)	1.707E+0(3.04E+0)
Exponential	100	**-1.000E+0(1.11E-16)**†	-1.000E+0(8.40E-7)	-1.000E+0(2.22E-16)
	300	**-1.000E+0(1.11E-16)**†	-1.000E+0(8.62E-8)	-1.000E+0(3.33E-16)
	500	**-1.000E+0(1.11E-16)**†	-1.000E+0(2.77E-8)	-1.000E+0(3.42E-9)
	1000	**-1.000E+0(8.88E-16)**†	-1.000E+0(6.78E-8)	-0.997E+0(5.24E-4)
Ackley	100	**1.465E-14(7.11E-15)**†	9.113E-3(1.38E-2)	5.296E-1(2.62E-1)
	300	**5.729E-14(1.24E-14)**†	8.972E-4(3.11E-3)	6.414E-7(2.59E-7)
	500	**3.504E-14(1.70E-13)**†	8.736E-4(2.81E-3)	5.769E-3(7.68E-4)
	1000	**7.224E-08(9.00E-8)**(2)†	7.211E-4(1.85E-3)	1.537E+0(6.42E-2)
Sphere	100	**0.000E+0(0.00E+0)**	4.022E-6(2.58E-5)	**0.000E+0(0.00E+0)**
	300	**0.000E+0(0.00E+0)**†	2.030E-6(2.37E-5)	3.913E-17(1.83E-17)
	500	**4.668E-27(1.86E-26)**†	4.091E-8(2.49E-6)	8.226E-7(2.02E-7)
	1000	**3.163E-15(9.31E-15)**†	2.718E-8(2.13E-7)	1.416E-1(1.66E02)
Styblinski Tank	100	**-3.917E+3(0.00E+0)**	-2.566e+3(6.97E+1)	**-3.917E+3(0.00E+0)**
	300	**-1.175E+5(0.00E+0)**†	-5.805E+3(1.48E+2)	-1.175E+4(2.93E-3)
	500	-1.958E+4(9.93E+3)	-8.721E+3(1.45E+2)	**-1.958E+4(3.90E-3)**
	1000	**-3.917E+4(3.91E-3)**†	1.550E+4(2.06E+2)	-3.913E+4(1.24E+1)
Rastrigin	100	3.41E+1(2.20E+1)	1.418E+0(7.82E+0)	**4.976E-1(9.95E-1)**‡
	300	2.980E+1(2.30E+1)	**2.441E-4(2.44E-3)**‡	1.992E+0(1.99E+0)
	500	9.770E-4(3.73E+1)	**0.000E+0(4.88E-4)**‡	1.614E+1(3.41E+0)
	1000	4.880E-3(9.77E-4)	**0.000E+0(0.00E+0)**‡	1.231E+3(7.33E+1)
Zakharov	100	2.240E+2(9.34E+1)	**9.584E+0(8.99E+0)**‡	2.099E+2(2.08E+2)
	300	7.755E+2(1.89E+2)	**5.623E+0(6.12E+0)**‡	6.302E+3(1.293E+3)
	500	1.424E+3(2.00E+2)	**2.625E+0(10.05E+0)**‡	1.255E+4(1.30E+3)
	1000	3.793E+3(3.66E+2)	**1.546E+0(4.77E+0)**‡	2.675E+4(2.20E+3)
Power Sum	100	**0.000E+0(0.00E+0)**	1.175e-7(2.13e-7)	**0.000E+0(0.00E+0)**
	300	**2.130E-43(1.91E-39)**†	2.2667E-8(5.964E-8)	7.460E-13(1.67E-12)
	500	**3.083E-32(1.23E-29)**†	1.157E-7(3.25E-8)	9.117E-8(5.52E-7)
	1000	**8.234E-25(1.86E-20)**†	3.724E-9(1.05E-8)	5.780E-5(6.04E-5)
Schwefel 2.21	100	**1.435E-10(1.97E-10)**†	1.755E-2(2.23E-2)	3.177E+0(1.49E-1)
	300	1.613E-1(3.36E-2)	**3.082E-4(3.49E-4)**‡	4.300E+0(8.29E-2)
	500	9.883E-1(1.12E-1)	**2.841E-5(5.56E-5)**‡	4.581E+0(7.61E-2)
	1000	2.660E+0(1.22E-1)	**4.419E-6(8.01E-6)**‡	4.812E+0(2.22E-2)

† denotes the performance of MTEA-RG is significantly better than the other peers according to the Wilcoxon's rank sum test at a 0.05 significance level; ‡ denotes the corresponding algorithm significantly outperforms MTEA-RG. Results show the Median and IQR over 30 independent runs.

Fig. 3: Comparison results on objective function value (median and IQR) for MTEA-RG($rmp = 0.3$) and other peer algorithms on suite of benchmark optimization problems.

in Section IV-A. The EMT-RE algorithm is implemented based on the authors experimental design with a population size $N + p = 100$, mating probability $rmp = 0.2$ and $T = 5$ tasks with distinct random embeddings. We set the embedding dimension $d_e = 30$ following the set-up used in [20] and place the box constraints $[-\sqrt{30}, \sqrt{30}]$ on the embedding space following the recommendation given in [21], [22]. We set-up the CC-DG-DE algorithm with a population size of 20 for each sub-problem, running the algorithm for 6 cycles of 50000 function evaluations each. For our comparison with the CC-DG-DE algorithm we do not take into account the number of evaluations required to carry out the differential grouping strategy.

The results provided in figure 3 show the proposed

MTEA-RG algorithm with $rmp = 0.3$ achieves better or equal performance over the other algorithms on 42 of the 56 benchmark problems. Both the EMT-RE and DE-DG algorithms perform better or equal on 10 of the 56 benchmark functions. The convergence plots in figure 2 also outline the restrictions of the EMT-RE and CC-DG-DE algorithms in comparison to the proposed MTEA-RG algorithm.

An interesting behaviour occurs on the optimization of the Zakharov function. The convergence graph shows both MTEA-RG and CC-DG-DE struggle to handle the flat search space of the Zakharov function throughout the optimization process. From this, we find that MTEA-RG also struggles to handle properties that have plagued the performance of evolutionary algorithm even for moderate dimension problems. In comparison the EMT-RE algorithm projects the problem into a space where its flat property is removed. In particular the optimum of the Zakharov function is the zero vector which is guaranteed to be within the embedded space as a result of the law of linear embeddings. In comparison, the convergence plot of the Styblinski-Tank function shows the restrictions of random embedding when the optimum of the function is away from the zero vector. As outlined by Letham et al. [23] imposing the box constraints removes the theoretical guarantee of a random embedding containing the optimum of a function, this issue is not addressed by the authors of the EMT-RE algorithm. In comparison, the MTEA-RG algorithm makes no transformation to the search space and still achieves competitive performance.

The obvious comparison with the MTEA-RG algorithm comes from the number of function evaluations required to carry out the differential grouping strategy. For benchmark functions this restriction has less impact due to their cheap evaluation. For real-world problems however. this restriction greatly limits the algorithm.

V. Conclusion and Future Directions

In this paper we propose the MTEA-RG algorithm for large-scale optimization. By decomposing an initial problem into several sub-problem using the random grouping strategy we show the positive impact knowledge transfer across sub-problems has on the optimization process. To illustrate the effectiveness of the proposed algorithm we carried out comparisons with the peer algorithms EMT-RE [20] and a differential grouping based algorithm CC-DG-DE on a suite of benchmark functions with varying dimensions. We show the positive impact sharing *cross-dimensional* information can have on the optimization performance. To the best of our knowledge this is the first work taking this perspective for large-scale optimization problems.

Considering a decomposed large-scale optimization problem as a multi-task optimization problems bring many areas of further research to improve its capability. Following the EMTO framework, an intuitive area of future directions of MTEA-RG algorithm is adaptability of the rmp to different sub-problems throughout the optimization or even the full removal of the rmp parameter by using machine learning models to select solutions for knowledge transfer. A simple direction to extend the algorithms performance is the use of different evolutionary algorithms and its conversion to an implicit knowledge transfer algorithm.

A limitation of the current MTEA-RG algorithm is its grouping, as variables of the same sub-problem cannot share information. Our future direction aims to address this issue by fully decomposing the problem at first and building up the dimension of sub-problems throughout the optimization process using machine learning models.

References

[1] M. Mitchell, *An introduction to genetic algorithms*, 1998.

[2] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," 2011.

[3] H. Beyer and H. Schwefel, "Evolution strategies - A comprehensive introduction," 2002.

[4] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," 2008.

[5] M. A. Potter and K. A. D. Jong, "A cooperative coevolutionary approach to function optimization," 1994.

[6] A. Gupta, Y. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," 2016.

[7] A. Slowik and H. Kwasnicka, "Evolutionary algorithms and their applications to engineering problems," 2020.

[8] X. Zheng, Y. Lei, A. K. Qin, D. Zhou, J. Shi, and M. Gong, "Differential evolutionary multi-task optimization." IEEE, 2019, pp. 1914–1921.

[9] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," 2004.

[10] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2008, June 1-6, 2008, Hong Kong, China*, 2008.

[11] M. N. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping for large scale non-separable function optimization," 2010.

[12] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," 2014.

[13] Y. Sun, M. Kirley, and S. K. Halgamuge, "Extended differential grouping for large scale global optimization with direct and indirect variable interactions," 2015.

[14] Y. Ling, H. Li, and B. Cao, "Cooperative co-evolution with graph-based differential grouping for large scale global optimization," 2016.

[15] M. N. Omidvar, M. Yang, Y. Mei, X. Li, and X. Yao, "Dg2: A faster and more accurate differential grouping for large-scale black-box optimization," 2017.

[16] Y. Wen and C. Ting, "Parting ways and reallocating resources in evolutionary multitasking," 2017.

[17] K. K. Bali, Y. Ong, A. Gupta, and P. S. Tan, "Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II," 2020.

[18] K. K. Bali, A. Gupta, L. Feng, Y. Ong, and P. S. Tan, "Linearized domain adaptation in evolutionary multitasking," 2017.

[19] L. Feng, W. Zhou, L. Zhou, S. W. Jiang, J. H. Zhong, B. S. Da, Z. X. Zhu, and Y. Wang, "An empirical study of multifactorial PSO and multifactorial DE," 2017.

[20] Y. Feng, L. Feng, Y. Hou, and K. C. Tan, "Large-scale optimization via evolutionary multitasking assisted random embedding," 2020.

[21] H. Wang, S. Rahnamayan, and Z. Wu, "Parallel differential evolution with self-adapting control parameters and generalized opposition-based learning for solving high-dimensional optimization problems," 2013.

[22] Z. Wang, F. Hutter, M. Zoghi, D. Matheson, and N. de Freitas, "Bayesian optimization in a billion dimensions via random embeddings," 2016.

[23] B. Letham, R. Calandra, A. Rai, and E. Bakshy, "Re-examining linear embeddings for high-dimensional bayesian optimization," 2020.

SUPPLEMENTARY MATERIAL

TABLE II: Comparison results on objective function value (median and IQR) for different rmp values of the MTEA-RG algorithm on suite of benchmark optimization problems

	d	0.0	0.1	0.3	0.5	0.7
Ackley	100	2.868E-6(5.83E-7)	2.176E-14(7.11E-15)	**1.465E-14(7.11E-15)**	4.942E-2(1.16E+0)	3.346E+0(1.80E+0)
	300	2.258E+0(9.26E-2	2.594E-4(1.54E-4)	5.729E-14(1.24E-14)	**3.952E-14(1.07E-14)**	1.496E-1(6.31E-1)
	500	5.211E+0(1.95E-1)	3.830E-2(1.77E-2)	3.504E-13(1.70E-13)	**6.795E-14(1.07E-14)**	6.09E-2(3.88E-1)
	1000	11.43E+0(1.87E-1)	3.37E+0(4.09E-1)	7.224E-08(9.0E-8)	**2.207E-13(3.02E-14)**	51.724E-9(2.93E-3)
Sphere	100	5.792E-13(1.64e-13)	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	1.115E-12(2.81E-7)	1.968E-1(6.68E-1)
	300	2.682E-1(2.98E-1)	3.892E-8(8.31E-8)	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	3.09E-3(2.58E-2)
	500	1.128E+1(1.02E+0)	5.723E-4(7.64E-4)	4.668E-27(1.86E-26)	**0.0E+0(0.0E+0)**	2.826E-10(1.96E-4)
	1000	2.688E+2(1.46E+1)	5.66E+0(2.29E+0)	3.163E-15(9.31E-15)	**0.0E+0(0.0E+0)**	1.227E-22(7.66E-13)
Styblinski Tank	100	**-3.917e+3(0.0E+0)**	**-3.917E+3(0.0E+0)**	**-3.917E+3(0.0E+0)**	-3.915E+3(8.32E+0)	5-3.864E+3(4.47E+1)
	300	-1.148E+4(7.02E+1)	**-1.175E+4(0.0E+0)**	**-1.175E+4(0.0E+0)**	**-1.175E+4(0.0E+0)**	**-1.175E+4(4.0E+0)**
	500	1.746E+4(8.20E+3)	-1.958E+4(9.953E+3)	-1.958E+4(9.932E+3)	-1.958E+4(9.952E+3)	**-1.958E+4(9.722E+3)**
	1000	-3.470E+4(1.85E+2)	-3.906E+4(6.47E+1)	-3.917E+4(3.91E-3)	**-3.917E+4(0.0E+0)**	-3.917E+4(2.93E-3)
Rastrigin	100	2.74E+2(1.50E+1)	**1.99E+1(1.74E+1)**	3.41E+1(2.20E+1)	45.88E+1(2.32E+1)	9.97E+1(4.90E+1)
	300	1.273E+3(3.37E+1)	**4.883E-4(2.44E-4)**	2.98E+1(2.98E+1)	3.88E+1(4.64E+1)	1.04E+2(6.10E+1)
	500	2.525E+3(5.22E+1)	1.07E-1(2.06E-1)	**9.77E-4(3.73E+1)**	3.26E-1(4.97E+1)	7.88E+1(6.24E+1)
	1000	6.430E+4(9.09e+1)	4.917E+2(2.56E+2)	**4.88E-3(9.77E-4)**	3.91E-3(9.77E-4)	1.021E+2(1.88E+2)
Zakharov	100	2.368E+2(1.37E+2)	**2.067E+2(1.04E+2)**	2.24E+2(9.34E+1)	2.28E+2(9.82E+1)	2.011E+2(1.21E+2)
	300	**6.659E+2(1.57E+2)**	7.218E+2(2.31E+2)	7.755E+2(1.89E+2)	8.553E+2(2.34E+2)	9.861E+2(2.758E+2)
	500	1.446E+3(2.86E+2)	**1.419E+3(2.62E+2)**	1.424E+3(2.00E+2)	1.633E+3(1.37E+2)	2.087E+3(2.71E+2)
	1000	4.555E+3(3.47E+2)	**3.986E+3(2.27E+2)**	3.793E+3(3.66E+2)	4.120E+3(4.06E+2)	5.287E+3(7.21E+2)
Power Sum	100	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	6.001E-13(8.603E-10)	1.458E-5(6.23E-4)
	300	**0.0E+0(0.0E+0)**	0.0E+0(3.15E-45)	2.130E-43(1.91E-39)	7.551E-28(1.97E-14)	1.475E-5(7.72E-4)
	500	**0.0E+0(4.412E-41)**	1.959E-41(6.30E-39)	3.083E-32(1.23E-29)	3.028E-29(8.87E-23)	6.947E-7(1.63E-5)
	1000	**1.034E-32(1.34E-30)**	9.583E-30(2.73E-28)	8.234E-25(1.86E-20)	6.899E-16(2.19E-14)	3.791E-8(2.13E-5)
Schwefel 2.21	100	2.744E-7(1.26E-7)	5.559E-9(3.05E-9)	**1.435E-10(1.97E-10)**	4.163E-1(5.11E-1)	2.911E+0(1.90E+0)
	300	1.931E-1(3.339E-2)	1.947E-1(4.18E-2)	**1.613E-1(3.36E-2)**	2.188E-1(5.96E-2)	2.088E+0(1.47E+0)
	500	9.473E-1(8.40E-2)	**9.433E-1(1.05E-1)**	9.883E-1(1.116E-1)	1.133E+0(1.40E+0)	2.858E+0(1.67E+0)
	1000	**2.483E+0(1.48E-1)**	2.519E+0(2.61E-1)	2.660E+0(1.22E-1)	2.871E+0(2.17E-1)	3.959E+0(7.05E-1)
Rosenbrock	100	**9.629E+1(5.60E+1)**	9.659E+1(4.57E-2)	9.698E+1(5.54E+1)	9.788E+1(3.62E+1)	1.538E+2(2.16E+2)
	300	3.715E+2(6.618E+1)	2.949E+2(1.37E-1)	2.95E+2(3.27E+2)	**1.062E-10(2.96E+2)**	2.460E+0 (2.96E+2)
	500	1.260E+3(1.159E+2)	4.948E+2(1.45E+0)	1.460E-2(4.94E+2)	**1.069E-08(8.33E-05)**	9.167E-07(4.94E+2)
	1000	1.827E+4(1.78E+3)	2.190E+3(5.60E+2)	3.238E+2(9.85E+2)	**2.28E-4(7.41E+2)**	2.495E-4(9.89E+2)
Ridge	100	-5.0E+0(4.77E-7)	**-5.00E+0(0.0)**	**-5.00E+0(0.0)**	-5.00E+0(6.91E-6)	-3.471E+0(3.62E+0)
	300	-4.488E+0(2.12E-2)	-5.00E+0(1.94E-4)	**-5.00E+0(4.77E-07)**	-5.00E+0(9.54E-7)	-3.123E+0(3.79E+0)
	500	-1.677E+0(1.78E-1)	-4.963E+0(1.45E-2)	**-5.00E+0(1.43E-6)**	**-5.00E+0(1.43E-6)**	-3.397E+0(3.80E+0)
	1000	11.03E+0(2.89E+0)	-2.578E+0(4.40E+0)	**-5.0E+0(4.16E-6)**	-5.0E+0(2.96E+0)	-6.562E-2(1.53E+0)
Schwefel	100	7.032E+3(1.326E+3)	**1.184E+3(2.73E+3)**	2.368E+3(1.302E+3)	2.697E+3(2.109E+3)	3.093E+3(1.985E+3)
	300	4.689E+4(9.02E+2)	3.989E-3(2.665E+3)	**3.818E-3(0.0E+0)**	3.818E-3(1.343E+2)	4.566E+2(3.761E+3)
	500	8.657E+4(1.03E+3)	2.888E+0(1.28E+1)	**6.364E-4(4.66E-10)**	**6.364E-4(4.66E-10)**	3.397E+2(8.92E+2)
	1000	1.895E+5(2.125E+3)	2.077E+4(1.07E+3)	1.273E-2(5.59E-9)	**1.273E-2(1.63E-9)**	5.984E-2(1.175E+1)
Alpine N1	100	3.636E-27(8.29E-27)	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	7.944E-7(1.64E+0)
	300	5.828E-6(2.96E-06)	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**
	500	1.432E-2(3.78E-3)	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**
	1000	1.196E+1(1.34E+1)	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**
Brown	100	1.320E-13(4.6E-14)	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	1.802E-14(5.75E-7)	7.749E-2(1.60E-1)
	300	5.033E-2(5.77E-3)	1.102E-8(1.53E-8)	**0.0E+0(0.0E+0)**	**0.0E+0(0.0E+0)**	1.285E-3(1.469E-2)
	500	1.925E+0(1.51E-1)	2.221E-4(2.0E-4)	8.716E-27(3.87E-26)	**0.0E+0(0.0E+0)**	3.290E-8(7.54E-4)
	1000	4.22E+1(2.65E+0)	1.118E+0(3.84E-1)	2.295E-15(5.10E-15)	**1.726E-31(5.49E-31)**	3.108E-20(1.86E-09)
Ackley N4	100	-2.90E+2(1.038E+0)	**-2.90E+2(6.1E-5)**	-2.804E+2(1.11E+1)	-2661E+2(2.07E+1)	-2.122E+2(6.81E+1)
	300	-1.60E+2(2.27E+1)	**-8.741E+2(9.0E-4)**	-8.742E+2(1.63E-1)	-8.735E+2(2.95E+1)	-8.300E+2(3.71E+1)
	500	5.797E+2(4.02E+2)	-1.457E+3(7.33E-1)	**-1.458E+3(9.86E-3)**	-1.457E+3(5.59E-2)	-1.394E+3(7.88E+1)
	1000	4.307E+3(9.18E+1)	-1.559E+3(4.70E+2)	**-2.917E+3(1.67E-1)**	-2.916E+3(7.38E-1)	-2.910E+3(1.03E+2)
Exponential	100	-1.0E+0(3.886E-15)	**-1.0E+0(1.11E-16)**	**-1.0E+0(1.11E-16)**	-1.0E+0(1.72E-9)	-9.954E-1(1.87E-2)
	300	9.948E-1(4.84E-4)	-1.0E+0(1.010E-9)	**-1.0E+0(1.11E-16)**	-1.0E+0(2.22E-16)	-1.0E+0(3.02E-4)
	500	-8.086E-1(2.03E-2)	-1.0E+0(1.74E-5)	**-1.0(1.11E-16)**	**-1.0E+0(1.11E-16)**	-1.0E+0(3.25E-4)
	1000	-5.365E-3(1.61E-3)	9.134E-1(3.59E-2)	-1.000E+0(8.88E-16)	**-1.000E+0(1.110E-16)**	-1.000E+0(1.38E-12)
Total Best:		8	20	**27**	23	5