Linearized Domain Adaptation in Evolutionary Multitasking

Kavitesh Kumar Bali*, Abhishek Gupta*, Liang Feng†, Yew Soon Ong*, Tan Puay Siew‡
* School of Computing Science and Engineering,
Nanyang Technological University, Singapore.
† College of Computer Science,
Chongqing University, China.
‡ Singapore Institute of Manufacturing Technology (SIMTech),
A*STAR, Singapore.

Abstract—**Recent analytical studies have revealed that in spite of promising success in problem solving, the performance of evolutionary multitasking deteriorates with decreasing similarity between constitutive tasks. The present day multifactorial evolutionary algorithm (MFEA) is susceptible to negative knowledge transfer between uncorrelated tasks. To alleviate this issue, we propose a linearized domain adaptation (LDA) strategy that transforms the search space of a simple task to the search space similar to its constitutive complex task. This high order representative space resembles high correlation with its constitutive task and provides a platform for efficient knowledge transfer via crossover. The proposed framework, LDA-MFEA is tested on several benchmark problems constituting of tasks with different degrees of similarities and intersecting global optima. Experimental results demonstrate competitive performances against MFEA and shows that our proposition dramatically improves the performance relative to optimizing each task independently.**

I. Introduction

Optimization is ubiquitous in our every day life. Evolutionary Algorithms (EA's) [1] are effective methods for solving optimization problems by using biologically inspired theory of *"Darwinian evolutionary system"* [2]. Recently, *evolutionary multitasking* has achieved significant success in solving multiple complex optimization problems simultaneously [3]. While most EA's are adept at solving a single optimization task, evolutionary multi-tasking exploits the implicit parallelism of a single population to solve multiple optimization tasks together in a *unified* search space via knowledge transfer. The knowledge (genetic materials) between tasks are transfered implicitly through chromosomal crossover [4]. This typically facilitates high quality convergence of the tasks being optimized in tandem, preferably when the fitness landscapes of distinct tasks are complementary [3, 5, 6]. The application domain of evolutionary multi-task optimization [3, 4] which focuses on problem solving is substantially distinguished from multi-task learning, that aims to *learn* tasks in parallel by sharing knowledge between them so as to improve their overall learning efficacy [7–10].

In literature, evolutionary multitasking has shown promising results, primarily when the tasks being optimized are correlated or share high magnitudes of similarity [4, 6]. Conversely, its performance deteriorates with low similarity between the constitutive tasks, suggesting that transfer of genetic materials between uncorrelated tasks can be detrimental. Negative transfer can eventually lead to inferior solution qualities and that the notion of knowledge transfer maybe be trivial in such scenarios. Therefore, it is imperative to facilitate positive transfer between uncorrelated tasks in the field of evolutionary multitask optimization.

Accordingly, we propose a novel idea of linearized domain adaptation (LDA) for effective problem solving using evolutionary multitasking. Domain adaptation is a fundamental aspect of machine learning [11, 12] and has been quite prevalent in the context of multi-task learning [8–10]. In this paper, we introduce a linear transformation strategy that maps the search space of a simpler task to the search space of a complex task. This induces a higher order representative search space which is highly correlated to that of the complex task. Herein, knowledge transfer between the tasks is executed efficiently while being optimized in concert. By leveraging on domain adaptation for knowledge transfer, we assert that the tasks in consideration need not be correlated and that our proposition is adept at solving both homogeneous and heterogeneous problem domains.

The organization of the rest of this paper is as follows. Section II describes the preliminaries and background of evolutionary multitasking. Sections III- IV describe the proposed method and its application to different categories of benchmark problems. Experimental results and their analysis are provided in Section V. Section VI concludes the paper with a brief discussion of future work.

II. Evolutionary Multitasking

Multifactorial Optimization (MFO) [3] is an evolutionary multitasking paradigm, characterized by the concurrent existence of multiple search spaces corresponding to different tasks that may or may not be correlated. Recently, Multifactorial Evolutionary Algorithm (MFEA) [3] has been introduced to solve multiple tasks concurrently using an evolutionary approach.

978-1-5090-4601-0/17/$31.00 ©2017 IEEE

A. Preliminaries

In MFO, each constitutive task is considered to impart an additional influence on the evolutionary process of a single population of individuals. Consider a case wherein K optimization tasks are to be performed concurrently. Herein all tasks are assumed to be minimization problems. The j^{th} task, denoted T_j, is considered to have a search space X_j on which the objective function is defined as $F_j : \boldsymbol{X}_j \rightarrow R$. In addition, each task may be constrained by several equality and/or inequality conditions that must be satisfied for a solution to be considered feasible. Accordingly, MFO is defined as an evolutionary multitasking paradigm that aims to simultaneously navigate the design space of all tasks, constantly building on the implicit parallelism of population-based search so as to rapidly workout $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{K-1}, \boldsymbol{x}_K\} = argmin\{F_1(\boldsymbol{x}), F_2(\boldsymbol{x}), \ldots, F_{K-1}(\boldsymbol{x}), F_K(\boldsymbol{x})\}$, where $\boldsymbol{x}_j$ is a feasible solution in $\boldsymbol{X}_j$. Each F_j is treated as an additional factor influencing the evolution of a single population of individuals. Thus, the composite problem may also be referred to as a K-factorial problem.

In MFO, a general technique for comparing population members in a multitasking environment is given as follows. Firstly, a set of properties for every individual p_i, where $i \in \{1, 2, , |P|\}$, in a population P is defined. Notably, the individuals are encoded in a unified search space $\boldsymbol{Y}$ encompassing $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_K$, a nd can be decoded into a task-specific solution representation with respect to each of the K optimization tasks. The decoded form of p_i can thus be expressed as $\{\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{iK}\}$, where $\boldsymbol{x}_{i1} \in \boldsymbol{X}_1$, $\boldsymbol{x}_{i2} \in \boldsymbol{X}_2$, $\ldots$, and $\boldsymbol{x}_{iK} \in \boldsymbol{X}_K$.

- *Definition 1*(*Factorial Cost*): For a given task T_j, the *factorial cost* Ψ_{ij} of individual p_i is given by $\Psi_{ij} = \lambda \cdot \delta_{ij} + F_{ij}$; where λ is a large penalizing multiplier, F_{ij} and δ_{ij} are the objective value and the total constraint violation, respectively, of p_i with respect to T_j. Accordingly, if p_i is feasible with respect to T_j (zero constraint violation), we have $\Psi_{ij} = F_{ij}$.
- *Definition 2*(*Factorial Rank*): The *factorial rank* r_{ij} of p_i on task T_j is simply the index of p_i in the list of population members sorted in ascending order with respect to factorial cost Ψ_{ij}.

While assigning factorial ranks, whenever $\Psi_{1j} = \Psi_{2j}$ for a pair of individuals p_1 and p_2, the parity is resolved by random tie-breaking.

- *Definition 3*(*Skill Factor*): The *skill factor* τ_i of p_i is the one task, amongst all other tasks in a K-factorial environment, with which the individual is associated. If p_i is evaluated for all K tasks then $\tau_i = argmin_j\{r_{ij}\}$, where $j \in \{1, 2, \ldots, K\}$.
- *Definition 4*(*Scalar Fitness*): The *scalar fitness* of p_i in a multitasking environment is given by $\varphi_i = 1/r_{iT}$, where $T = \tau_i$ (r_{iT} is the best overall rank over all tasks). Note that $max\{\varphi_i\} = 1$.

Subsequently, once the fitness of every individual has been *scalarized* (Definition 4), performance comparison is executed in a straightforward manner. For instance, individual p_1 will be considered to dominate individual p_2 in multifactorial sense simply if $\varphi_1 > \varphi_2$.

Since the factorial rank of an individual, and implicitly its scalar fitness, depends on the performance of every other individual in the population, the comparison is in fact population dependent. Nevertheless, the procedure guarantees that if an individual p^* attains the global optimum of any task then $\varphi^* = 1$, which implies that $\varphi^* \geq \varphi_i$ for all $i \in \{1, 2, \ldots, |P|\}$.

- *Definition 5*(*Multifactorial Optimality*): An individual p^* is considered to be optimum in multifactorial sense if there exists at least one task in the K-factorial environment which it globally optimizes.

B. Search Space Unification

The notion of search space unification implies that genetic building blocks [13] corresponding to different tasks are contained within a single pool of genetic material. This process ideally facilitates faster parallel processing. To elaborate, assuming the search space dimensionality of the j^{th} optimization task (in isolation) to be D_j, a unified search space $\boldsymbol{Y}$ comprising K (traditionally distinct) tasks may be defined such that $D_{multitask} = max_j\{D_j\}$, where $j \in \{1, 2, \ldots, K\}$. That is, while dealing with K optimization tasks simultaneously, the chromosome $\boldsymbol{y} \in \boldsymbol{Y}$ of an individual in the MFEA is represented by a vector of $D_{multitask}$ variables. While addressing the j^{th} task, D_j variables are extracted from the chromosome and decoded into a meaningful solution representation for the associated optimization task using a random-key unification scheme [14]. By the definition of random-key scheme, each variable of a chromosome is simply encoded by a *continuous value* in the range $[0, 1]$. The decoded representation is achieved by linearly mapping each random-key from the genotype space to the bound-constrained phenotype space (upper and lower bounds) of the relevant continuous optimization task [3].

C. Multifactorial Evolutionary Algorithm

In this section we briefly describe the present day MFEA [6] which is an effective multitasking framework inspired by the bio-cultural models of multifactorial inheritance [5]. Traditional evolutionary algorithms utilize a large population of candidate solutions, all of which may not be suited for the task at hand. Conversely, in a multitasking environment, a randomly generated or evolved (genetically modified) individual is competent for at least one task of the many tasks in consideration. Accordingly, MFEA effectively splits the population into different skill factors, each excelling at a different task. Moreover, during the evolutionary process, it is possible that the genetic material created within a particular task might be useful to another task. Under such circumstances, transfer of genetic materials between tasks is considered to be ideal in order to improve the overall optimization process. In MFEA, the transfer of genetic material is facilitated via crossover between individuals of different tasks (skill factors). It should be noted that the genetic transfer is implicit and is achieved by, 1.) a random probability (rmp) which permits individuals of different tasks to crossover and 2.) selective imitation wherein the generated offspring can randomly select a skill factor corresponding to either of the parent. Authors in [3] present a trade-off that *"while excessive communication can be disruptive to focused search, prohibiting communication is also undesirable as it constrains exploration and the power of implicit parallelism offered by the entire population"*. The interested reader is referred to [3].

III. Linearized Domain Adaptation

Evolutionary multitasking exploits a single population of individuals to solve multiple tasks in concert. For the sake of brevity, consider two tasks T_1 and T_2 of dimensions D_1, D_2 corresponding to objective functions f_1 and f_2. To ensure unbiased representation of all optimization tasks, MFEA initially assigns approximately similar magnitude of individuals to each of the constitutive tasks.

Accordingly, consider two matrices $A(m \times D_1)$ and $B(n \times D_2)$. The matrix A contains all the solutions corresponding to task T_1 of dimensions D_1 and B corresponds to all the solutions of task T_2 of dimensions D_2.

A. Ordinal Correlation

The significance of ordinal correlation in the context of evolutionary multitasking has been demonstrated in [4]. To elaborate, multitask optimization is guaranteed to work if two (minimization) tasks T_1 and T_2 with cost/objective functions f_1 and f_2 share high ordinal correlation (high similarity). Herein the assumption is that for any given pair of solutions u_i and u_j in the unified search space, $f_1(u_{i1}) < f_1(u_{j1}) \Leftrightarrow f_2(u_{i2}) < f_2(u_{j2})$. This implies that by coupling the two tasks in a multitasking environment, any steps leading to a cost reduction (fitness improvement) for T_1 will automatically lead to a cost reduction for T_2 for free, and vice versa, with out consuming any extra fitness evaluations. Thus, multitask optimization bestows the scope for free lunches [15] with functions characterized by high ordinal correlation.

Howsoever, this condition may not hold true in all cases. In general, a pair of tasks may not always satisfy the above theorem especially if they are characterized by functions of low similarity/correlation. Therefore, in order to facilitate high ordinal correlations between the tasks, we transform the search space of at least one task so that the resultant space shares high correlation with its constitutive task. Accordingly, matrices A and B are sorted (ranked) in ascending order according to the fitness values obtained by f_1 and f_2. The process of sorting (ranking) of solutions is what actually facilitates the ordinal correlation between the two data sets A and B . Next, a linear transformation M is computed that transforms the search space of A to B. Further details of the linear mapping is provided in the next section.

B. Linear Transformation

Suppose we wish to find a linear transformation M that maps A to B such that it satisfies the principle of least square mapping. Based on the following relation:
$A \cdot M = B,$ which implies that: $(A^TA)M = A^TB$

$$\text{Thus, } M = (A^TA)^{-1}A^TB \tag{1}$$

Conversely, the reverse mapping M' can be derived by the following steps:

Considering $A \cdot M = B$,

$$AMM^T = BM^T$$
$$A = BM^T(MM^T)^{-1}$$
$$\text{Therefore, } M' = M^T(MM^T)^{-1} \tag{2}$$

Therefore, given the mappings M and M', we can transform the search space of A to B as follows:
$A' = A.M$ such that $A' \simeq B$, implying that the transformed high order representative space A' of task T_1 is highly correlated (similar) to the search space of task T_2 represented by B. On the flip side, A' can be reverted back to its original space via reverse transformation: $A'.M' = A$. The linearized domain adaptation strategy is summarized in Algorithm 1.

Algorithm 1: $(M,M')=mapping(A,B)$

```
1 Stage 1: Sort and Choose most representative points of $A$ and $B$
2 Stage 2: Compute Forward Mapping: A → B
3         $M = (A^TA)^{-1}A^TB$          (1)
4 Stage 3: Compute Reverse Mapping: B → A
5         $M' = M^T(MM^T)^{-1}$          (2)
```

C. Handling Scalability Issues

Since the data points accumulated in A and B gradually increases at every generation and can potentially be very large , we utilize k-means clustering to select the best representative points of each, i.e k cluster centroids. The parameter k defining the number of clusters is iteratively increased by a predefined step size at every generation. Subsequently, the dimensions of matrices A and B are rescaled to $(k \times D_1) and (k \times D_2)$ respectively and an efficient mapping strategy is ensued.

IV. Enhanced LDA-MFEA

This section describes an enhanced multifactorial evolutionary algorithm that features a linearized domain adaptation strategy (LDA) for efficient problem solving. We label this variant of evolutionary multitasking as LDA-MFEA. Notably, we eliminate mutation and study the effect knowledge transfer purely through crossover using domain adaptation.

Algorithm 2 describes the basic structure of LDA-MFEA . It starts by randomly initializing a population P_0 of n individuals all of which are pooled in a unified search space Y (line 1). Since the goal is to solve multiple tasks concurrently, each individual (of the initial population) is assigned a skill factor corresponding to a particular task. Initially, each task has equal number of representatives so as to ensure unbiased setup. The pre-assigned skill factor of an individual is the computer analogous of the pre-assigned cultural trait of the natural phenomenon. Accordingly, the individuals of P_0 are

Algorithm 2: Pseudo-code of the LDA-MFEA

```
1: Randomly generate $n$ individuals in $\boldsymbol{Y}$ to form initial population $P_0$
2: for every $p_j$ in $P_0$ do
3:     Assign skill factor $\tau_j = \mod(j, K) + 1$, for the case of $K$ tasks
4:     Evaluate $p_j$ for task $\tau_j$ only
5: end for
6: Compute scalar fitness $\varphi_j$ for every $p_j$
7: Set $t = 0$
8: while stopping conditions are not satisfied do
9:     for every $p_j$ in $P_t$ do
10:        Accumulate $p_j$ of skill factor $\tau = 1$ into matrix $A$
11:        Accumulate $p_j$ of skill factor $\tau = 2$ into matrix $B$
12:    end for
13:    Compute mappings: $[M, M'] = mapping(A, B) \rightarrow$ Refer Algorithm 1
14:    $C_t$ = LDA + Crossover($P_t$) $\rightarrow$ Refer Algorithm 3
15:    for every $c_j$ in $C_t$ do
16:        Determine skill factor $\tau_j \rightarrow$ Refer Algorithm 4
17:        Evaluate $c_j$ for task $\tau_j$ only
18:    end for
19:    $R_t = C_t \cup P_t$
20:    Update scalar fitness of all individuals in $R_t$
21:    Select $N$ fittest members from $R_t$ to form $P_{t+1}$
22:    Set $t = t + 1$
23: end while
```

Algorithm 3: Linearized domain adaptation and Crossover

```
1  Randomly select two parent candidates $p_a$ and $p_b$ from $P_t$
2  Generate a random number $rand$ between 0 and 1
3  if $\tau_a == \tau_b$ or $rand \leq rmp$ then
4      Crossover
5      Parents $p_a$ and $p_b$ crossover to produce two offspring $c_1, c_2 \in C(t)$
6  if $\tau_a \neq \tau_b$ then
7      LDA + Crossover
8      if $\tau_a = 1$ then
9          Transform $p_a$ to higher order (task 2) search space via $M$
10         $p'_a = p_a * M$
11         Parents $p'_a$ and $p_b$ crossover to produce two offspring $c_1, c_2 \in C(t)$
12     if $\tau_b = 1$ then
13         Transform $p_b$ to higher order (task 2) search space via $M$
14         $p'_b = p_b * M$
15         Parents $p'_b$ and $p_a$ crossover to produce two offspring $c_1, c_2 \in C(t)$
```

evaluated only with respect to the assigned task. This practically conserves a huge amount of computing resources which would be consumed if all the individuals are evaluated across all the tasks available (lines 2-5). The scalar fitness of all the individuals of P_0 is computed after task-specific evaluations (line 6). Line 7 sets the initial generation count to zero and the loop on line 8 forms the main evolutionary loop.

To explain the mechanism of LDA-MFEA concisely, we consider a two task optimization scenario. Without loss of generality, task T_1 ($\tau = 1$) is considered to be the simpler problem and task T_2 ($\tau = 2$) being the complex one in the heterogeneous multitasking environment. Herein, our aim is to transform the search space of the simple task to that of the complex task and facilitate knowledge transfer (via crossover) in the highly correlated space. In order to find a descent linear mapping between the two tasks, the algorithm accumulates the chromosome vectors (points) associated with Tasks T_1 and T_2 and stores them into matrices A and B at every generation t (Alg. 2, lines 9-12). Accordingly, the forward and reverse linear mappings (M and M') are computed. The mapping strategy has been thoroughly described in section III.

Algorithm 4: Selective imitation and Reverse mapping

```
1  Consider offspring $c \in C_t$ where $c$ is result of Crossover or LDA + Crossover
2  if Crossover $\rightarrow$ Refer Algorithm 3 then
3      Generate a random number $rand$ between 0 and 1
4      if $rand \leq 0.5$ then
5          $c$ imitates skill factor of $p_a$
6      else
7          $c$ imitates skill factor of $p_b$
8  if LDA + Crossover $\rightarrow$ Refer Algorithm 3 then
9      Generate a random number $rand$ between 0 and 1
10     if $\tau_a = 1$ then
11         if $rand \leq 0.5$ then
12             $c$ imitates skill factor of $p_a$
13             $c = c * M'$
14         else
15             $c$ imitates skill factor of $p_b$
16     if $\tau_b = 1$ then
17         if $rand \leq 0.5$ then
18             $c$ imitates skill factor of $p_b$
19             $c = c * M'$
20         else
21             $c$ imitates skill factor of $p_a$
```

A. Crossover

The crossover mechanism of LDA-MFEA (Alg. 2, line 14 is described in Algorithm 3. It should be noted that the genetic operators associated with LDA-MFEA is only crossover without any mutation. Suppose two individuals p_a and p_b are randomly chosen for crossover. Lines 1-3 (Alg. 3) abide by the assortative mating principle wherein individuals prefer to mate with same cultural background. That is, individuals undergo crossover if they possess the same skill factor (τ_a == τ_b) implying they appeal to the same task. Additionally, the individuals may also undergo cross-cultural crossover according to some random probability (rmp).

Lines 5-16 of Algorithm 3 illustrate the linearized domain adaptation strategy employed in LDA-MFEA. In this case, the selected individuals p_a and p_b have different skill factors ($\tau_a \neq \tau_b$) implying that they correspond to different tasks. Consider a case whereby task T_1 being a simple task is to be mapped to a more complex task T_2 . Suppose that the randomly selected individual p_a (line 1) corresponds to task T_1 i.e $\tau_a = 1$ (line 7). Correspondingly, p_a is transformed to a higher order representative space p'_a which is highly correlated with the search space of p_b (task T_2 individual). Subsequently, p'_a and p_b crossover to produce two offspring individuals c_1 and c_2 (Alg. 3, lines 6-11). Alternatively, p_b undergoes transformation and crossover (Alg. 3,lines 12-16).

B. Selective imitation and Reverse mapping

Across each generation t of LDA-MFEA, it is practical to evaluate individuals for only the task that it appeals to (performs better). This idea is inspired from the memetic concept of vertical cultural transmission whereby an offspring is highly likely to experience strong cultural influences (e.g cultural traits) from either of its parents [16, 17]. That is, the phenotype of the parents naturally affect the phenotype of the child. Thus, it is common that a child imitates cultural behavior of its parents. The computer analogous of vertical cultural transmission in LDA-MFEA is provided by Algorithm 4 which describes the concept of selective imitation associated with the skill factor. Lines 2-7 (Alg. 4) shows a simple strategy that randomly assigns which parent the child imitates i.e, either of the skill factors τ_a or τ_b corresponding to p_a and p_b respectively. In the case wherein linearized domain adaptation is used to facilitate crossover in the highly correlated transformed space(Alg. 4,lines 8-21), the generated offspring c is mapped backed to its original task via the reverse mapping M'. The reverse mapping is entirely dependent on the skill factor an offspring c imitates. For instance, we refer back to Algorithm 3 (lines 8-11) wherein the individual p_a corresponding to task $T_1(\tau_a = 1)$ is transformed to higher order representative space p'_a. Subsequently, parents p'_a and p_b crossover to produce a pair offspring in $C(t)$. Since an offspring $c \in C(t)$ imitates p_a (by chance), c is transformed back to its original task specific search space (task T_1) and acquired in the unified search space Y (Alg. 4,lines 10-13). The reverse mapping M' ensures that the knowledge transfered in the highly correlated transformed space is transfered back to the original task. In LDA-MFEA, an offspring $c \in C(t)$ is evaluated for only one task that it imitates thus saving enormous computational expenses (Alg. 2, line 17).

Lines 19-22 of Algorithm 2 show that LDA-MFEA abides by an elitist strategy which ensures that the fittest individuals survive through the generations.

V. Experimental Results

In this section, we showcase the efficacy of the proposed LDA-MFEA by presenting experimental studies based on the synthetic benchmark functions [4] provided by Table I. These composite benchmark problems are classified into nine different categories based on the degree of intersection of the global optima of the constitutive optimization tasks and the levels similarity between them. The degrees of intersection are complete intersection (CI), partial intersection (PI), and no intersection (NI). The inter-task similarity measure between a pair of tasks (R_s) is based on Spearmans rank correlation similarity metric. Accordingly, the problem pair with $R_s < 0.2$ is considered to have low similarity, the problem pair with $0.2 \leq R_s < 0.8$ resembles medium similarity, and the problem pair with $R_s \geq 0.8$ implies high similarity/correlation. The dimensions of each of the tasks is represented by D_T. The performance of the LDA-MFEA (which executes multitasking) shall be compared against the current MFEA and its standalone base algorithm, i.e., the SOEA, that optimizes a single task at a time.

A. Parameter Setting

For all the algorithms under consideration, we switch off mutation and employ only the Simulated Binary Crossover (SBX) genetic operator [18] and maintain a consistent experimental setup. A population size of 100 individuals is evolved for a maximum of 500 generations for each task or until the solutions converge to the global optimums of the constitutive tasks. Additionally, every individual in LDA-MFEA, MFEA and SOEA undergoes a learning step. The individual learning is carried out via the BFGS quasi-Newton method abiding by the phenomenon of Lamarckism [19]. It should be noted that in LDA-MFEA and MFEA, a generated offspring undergoes local improvement only with respect to the skill factor that it imitates. The value of rmp is set to a tiny magnitude of 0.3 in LDA-MFEA (to promote frequent domain adaptation and efficient knowledge transfer). In MFEA [3], rmp is set to 1.0.

TABLE I
Benchmark problem sets [4]

Category	Task	D_T	Landscape	R_s
$_1$ CI+HS	Griewank (T_1)	50	multimodal, nonseparable	1.0000
	Rastrigin (T_2)	50	multimodal, nonseparable	
$_2$ CI+MS	Ackley (T_1)	50	multimodal, nonseparable	0.2261
	Rastrigin (T_2)	50	multimodal, nonseparable	
$_3$ CI+LS	Ackley (T_1)	50	multimodal, nonseparable	0.0002
	Schwefel (T_2)	50	multimodal, separable	
$_4$ PI+HS	Rastrigin (T_1)	50	multimodal, nonseparable	0.8670
	Sphere (T_2)	50	unimodal, separable	
$_5$ PI+MS	Ackley (T_1)	50	multimodal, nonseparable	0.2154
	Rosenbrock (T_2)	50	multimodal, nonseparable	
$_6$ PI+LS	Ackley (T_1)	50	multimodal, nonseparable	0.0725
	Weierstrass (T_2)	25	multimodal, nonseparable	
$_7$ NI+HS	Rosenbrock (T_1)	50	multimodal, nonseparable	0.9434
	Rastrigin (T_2)	50	multimodal, nonseparable	
$_8$ NI+MS	Griewank (T_1)	50	multimodal, nonseparable	0.3669
	Weierstrass (T_2)	50	multimodal, nonseparable	
$_9$ NI+LS	Rastrigin (T_1)	50	multimodal, nonseparable	0.0016
	Schwefel (T_2)	50	multimodal, separable	

B. Results

The results provided in Table II show the mean and the standard deviation of the final results obtained by 30 independent runs. Best mean values are highlighted bold. We first evaluate the performance of LDA-MFEA against the current MFEA. According to Table II, it can be observed that LDA-MFEA has indeed demonstrated competitive performance with respect to MFEA. Both LDA-MFEA and MFEA have performed equally well in categories $_1$ and $_2$ that are defined by complete intersection of the global optimums with high and low similarities between task pairs respectively. It is quite clear that LDA-MFEA has outperformed MFEA on category $_3$ in which the constitutive tasks share complete intersection and low similarity. In the case of partial intersection of the global optimum of the constitutive tasks (categories $_4-_6$), LDA-MFEA has performed better in the category sets that constitute of tasks with low and medium similarities (categories $_5$,$_6$). The efficacy of LDA-MFEA is highly evident in the heterogeneous problem set of category $_6$. Herein, T_2 of dimension size of 25 (simple task) is mapped to the search space of T_1 of dimension size 50 (complex task) so as to ensure high correlations between them. This essentially facilitates efficient knowledge transfer and ensure superior quality solutions for both tasks. It is noted that initially the tasks in category $_6$ share low similarity as shown in Table I and by leveraging domain adaptation, LDA-MFEA improves the performance of MFEA on the same. For the cases of no intersection between the global optimums of the tasks (categories $_7-_9$), it can be observed that both LDA-MFEA and MFEA performed remarkably well on T_2 of category $_7$ and in category $_8$, MFEA recorded slightly better solutions for both Tasks. In the extreme uncorrelated case of category $_9$, LDA-MFEA converged to a high quality solution for T_2 but was outperformed by MFEA for T_1.

While evaluating the performance of LDA-MFEA against SOEA, it is clear that LDA-MFEA substantially improves the performance relative to optimizing each task independently (SOEA) in most of the cases. According to Table II, LDA-MFEA has generally outperformed SOEA in categories $_1$-$_7$. On a closer look, one can observe that LDA-MFEA has recorded superior quality solutions on problem sets with low degree of similarities for categories $_3$ and $_6$ with complete and partial intersections respectively. For the extreme cases of no intersection between the global optimum of the tasks, LDA-MFEA has also improved the solution qualities in categories $_7$ (T_1) and $_8$ (T_2). In category $_9$, SOEA recorded better average solution for T_1 but LDA-MFEA clearly outperformed SOEA on T_2.

To support our contention that LDA-MFEA converges rapidly than SOEA, we provide convergence plots representing different levels of intersection and similarity. Figures 1 and 2 shows four representative convergence plots corresponding to categories $_1$,$_3$,$_6$,$_9$. In general, it is quite clear that LDA-MFEA has accelerated convergence in contrast to SOEA. In addition to converging remarkably fast with complete intersection and highly similar (HS) tasks (Figures 1(a), 1(b)), LDA-MFEA also converges faster than SOEA in the case with partial intersection (PI) and low similarity (LS) as shown by Figures 2(a), 2(b). In the extreme task pair of no intersection and low similarity (low correlation), Figure 2(d) shows that LDA-MFEA has better convergence than SOEA on T_2 which happens to be the complex task. These convergence characteristics suggests that the linearized domain adaptation can potentially alleviate the need for the tasks to be correlated whilst promoting efficient knowledge transfer.

VI. Conclusion

In this paper, we proposed a linearized domain adaptation (LDA) strategy that induces highly correlated search spaces to promote efficient knowledge transfer between distinct tasks. The results have shown that LDA-MFEA outperforms single task optimizers and also improves the performance of the present day MFEA [3]. We also show that LDA-MFEA is adept at solving both homogeneous and heterogeneous problem domains. To conclude, our future conjecture is that LDA-MFEA can potentially be extended to wide range of real world problems.

Acknowledgment

This work is supported by the Singapore Institute of Manufacturing Technology-Nanyang Technological University (SIMTech-NTU) Joint Lab on Complex Systems.

References

[1] T. Bäck, D. B. Fogel, and Z. Michalewicz, "Handbook of evolutionary computation," *New York: Oxford*, 1997.

[2] K. A. De Jong, *Evolutionary computation: a unified approach*. MIT press, 2006.

[3] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: toward evolutionary multitasking," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 343–357, 2016.

[4] B. Da, Y.-S. Ong, L. Feng, A. Qin, A. Gupta, Z. Zhu, C.-K. Ting, K. Tang, and X. Yao, "Evolutionary multitasking for single-objective continuous optimization: Benchmark problems, performance metric, and baseline results," Nanyang Technological University, Singapore, Tech. Rep., 2016, http://www.cil.ntu.edu.sg/mfo/download.html.

[5] Y.-S. Ong and A. Gupta, "Evolutionary multitasking: a computer science view of cognitive multitasking," *Cognitive Computation*, vol. 8, no. 2, pp. 125–142, 2016.

[6] A. Gupta, Y. Ong, B. Da, L. Feng, and S. Handoko, "Measuring complementarity between function landscapes in evolutionary multitasking," in *2016 IEEE Congress on Evolutionary Computation, accepted*.

[7] R. Caruana, "Multitask learning," in *Learning to learn*. Springer, 1998, pp. 95–133.

[8] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," *Advances in neural information processing systems*, vol. 19, p. 41, 2007.

[9] T. Jebara, "Multi-task feature and kernel selection for svms," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 55.

[10] U. Rückert and S. Kramer, "Kernel-based inductive transfer," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2008, pp. 220–233.

[11] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[12] J. Jiang, "A literature survey on domain adaptation of statistical classifiers," *URL: http://sifaka. cs. uiuc. edu/jiang4/domainadaptation/survey*, vol. 3, 2008.

[13] M. Iqbal, W. N. Browne, and M. Zhang, "Reusing building blocks of extracted knowledge to solve complex, large-scale boolean problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 465–480, 2014.

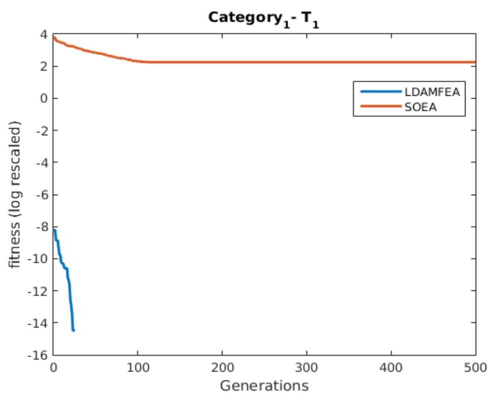
[14] J. C. Bean, "Genetic algorithms and random keys for sequencing and optimization," *ORSA journal on computing*, vol. 6, no. 2, pp. 154–160, 1994.

[15] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.

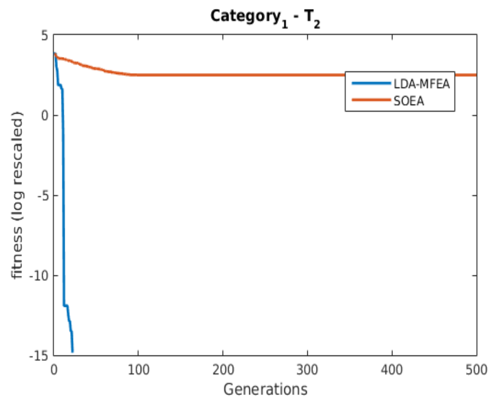
[16] X. Chen, Y.-S. Ong, M.-H. Lim, and K. C. Tan, "A multi-facet survey on memetic computation," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 5, pp.

TABLE II
AVERAGE PERFORMANCES (MEAN AND STANDARD DEVIATION) OF LDA-MFEA, MFEA AND SOEA ON DIFFERENT TASKS FOR 30 INDEPENDENT RUNS

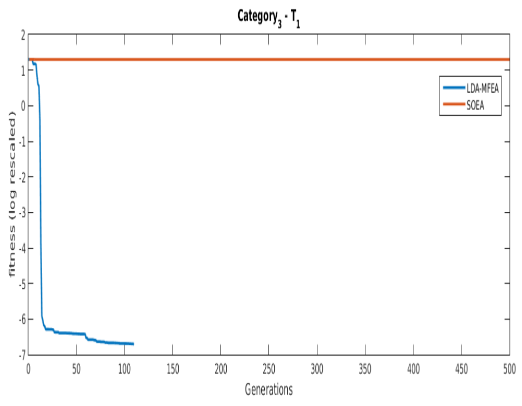
Category	LDA-MFEA		MFEA		SOEA	
	T_1	T_2	T_1	T_2	T_1	T_2
$_1$ CI + HS	**0.00** 0.00	**0.00** 0.00	**0.00** 0.00	**0.00** 0.00	1.67e-13 6.35e-14	220.88 57.48
$_2$ CI + MS	**3.28e-14** 1.01e-15	**0.00** 0.00	**3.14e-14** 1.75e-15	**0.00** 0.00	15.26 7.30	225.855 31.05
$_3$ CI + LS	**2.97e-07** 2.25e-07	**6.34e-04** 6.34e-04	19.98 0.02	1598.91 336.04	20.03 1.49e-03	3115.47 425.927
$_4$ PI + HS	125.23 22.50	**1.15e-26** 2.03e-26	**72.36** 11.18	1.76e-13 3.88e-14	209.74 27.91	1.56e-15 8.01e-16
$_5$ PI + MS	**9.82e-09** 1.05e-08	15.17 9.16	1.09e-08 2.29e-09	11.71 1.21	15.61 7.43	**6.34e-07** 1.27e-06
$_6$ PI + LS	**3.28e-24** 1.67e-24	**0.00** 0.00	4.14e-14 3.27e-14	0.96 2.87	17.18 5.75	18.39 2.47
$_7$ NI + HS	15.12 5.36	**0.00** 0.00	10.62 1.06	0.00 0.00	**1.21e-04** 2.34e-04	210.34 24.20
$_8$ NI + MS	3.06e-12 1.23e-12	24.81 2.76	2.80e-12 1.01e-12	**21.87** 2.19	**3.74e-13** 2.04e-13	44.78 3.18
$_9$ NI + LS	451.18 75.55	**6.36e-04** 1.03e-04	**68.85** 11.90	2060.83 309.76	220.98 29.75	2830.26 497.43



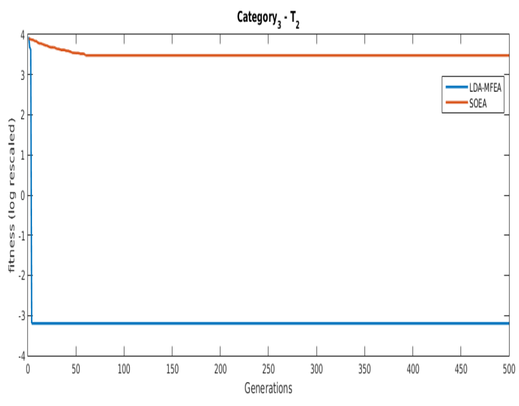
(a) $_1$ CI+HS T$_1$



(b) $_1$ CI+HS T$_2$

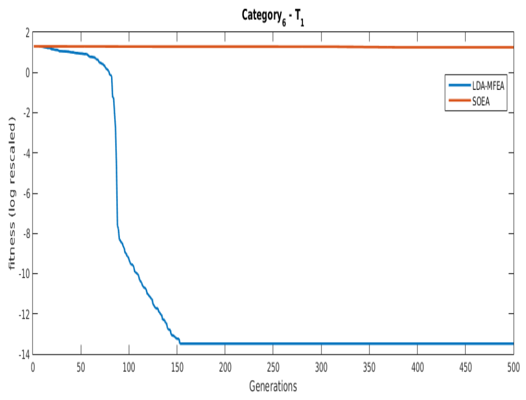


(c) $_3$ CI+LS T$_1$

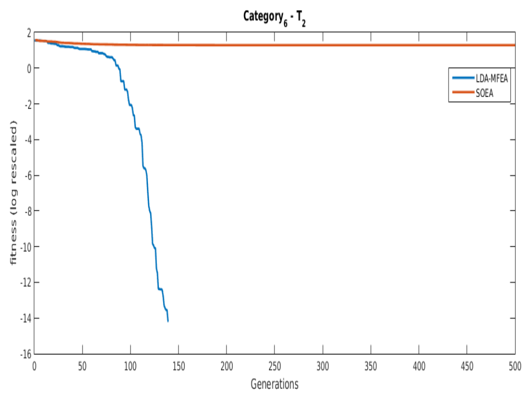


(d) $_3$ CI+LS T$_2$

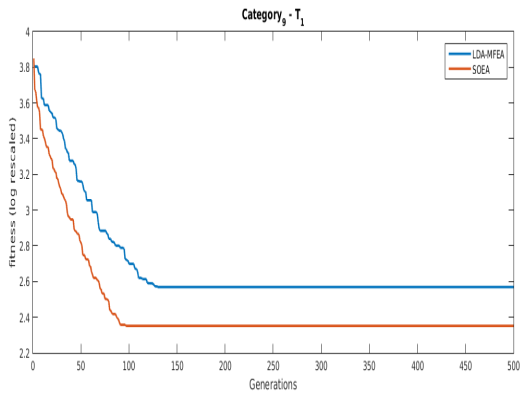
Fig. 1. Convergence plots of categories $_1$,and $_3$



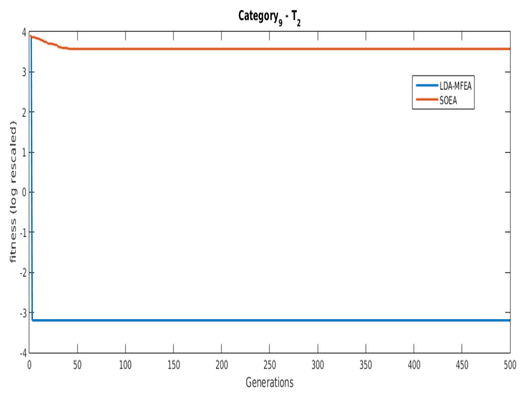
(a) $_6$ PI+LS T_1



(b) $_6$ PI+LS T_2



(c) $_9$ NI+LS T_1



(d) $_9$ NI+LS T_2

Fig. 2. Convergence plots of categories $_6$ and $_9$

591–607, 2011.

[17] M. W. Feldman and K. N. Laland, "Gene-culture coevolutionary theory," *Trends in Ecology & Evolution*, vol. 11, no. 11, pp. 453–457, 1996.

[18] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 3, pp. 1–15, 1994.

[19] Y. S. Ong and A. J. Keane, "Meta-lamarckian learning in memetic algorithms," *IEEE transactions on evolutionary computation*, vol. 8, no. 2, pp. 99–110, 2004.