

# Multifactorial Evolutionary Algorithm with Online Transfer Parameter Estimation: MFEA-II

\* Kavitesh Kumar Bali, \* Yew-Soon Ong, † Abhishek Gupta, † Puay Siew Tan

\* School of Computing Science and Engineering,  
Nanyang Technological University, Singapore.

† Singapore Institute of Manufacturing Technology (SIMTech),  
A\*STAR, Singapore.

**Abstract**—Humans rarely tackle every problem from scratch. Given this observation, the motivation for the present work is to improve optimization performance through adaptive knowledge transfer across related problems. The scope for spontaneous transfers under the *simultaneous* occurrence of multiple problems unveils the benefits of multitasking. Multitask optimization has recently demonstrated competence in solving multiple (related) optimization tasks concurrently. Notably, in the presence of underlying relationships between problems, the transfer of high-quality solutions across them has shown to facilitate superior performance characteristics. However, in the absence of any prior knowledge about the inter-task synergies (as is often the case with general black-box optimization), the threat of predominantly *negative transfer* prevails. Susceptibility to negative inter-task interactions can impede the overall convergence behavior. To allay such fears, in this paper, we propose a novel evolutionary computation framework that enables *online learning* and exploitation of the similarities (and discrepancies) between distinct tasks in multitask settings, for an enhanced optimization process. Our proposal is based on principled theoretical arguments that seek to minimize the tendency of harmful interactions between tasks, based on a purely *data-driven* learning of relationships among them. The efficacy of our proposed method is validated experimentally on a series of synthetic benchmarks, as well as a practical study that provides insights into the behavior of the method in the face of several tasks occurring at once.

**Index Terms**—Evolutionary Multitasking, multifactorial optimization, inter-task synergies, online similarity learning.

## I. INTRODUCTION

In machine learning, the notion of knowledge transfer has manifested in many guises, ranging from *transfer learning* [1–5] to *multitask learning* [6]. Transfer learning leverages on a pool of available data from various source tasks to improve the learning efficacy of a related target task. Interestingly, while transfer learning mainly focuses on just the target task, multitask learning, in contrast, learns all tasks simultaneously by sharing information across the different tasks, in order to improve the *overall* generalization performance. Simply put: *what is learned for each task can help other tasks be learned better*. Fueled by data abundance, the concepts of knowledge transfer have mainly been prominent in the field of predictive analytics (e.g. classification and regression). On the other hand, attempts made to transfer knowledge in the context of *optimization* problems (which typically start the search with *zero* prior data) [7], have relatively been scarce, and received little visibility.

However, it is contended that real world optimization problems seldom occur in isolation. That is, the presence of latent similarities, even between distinct black-box optimization instances, are commonplace. This is particularly true when we are working within a particular domain of application. With this in mind, a plethora of information can potentially be shared across related problems. Arguably, it may seem unnecessarily laborious to isolate and solve any such related optimization problems from scratch (i.e., from a zero-state knowledge) [7]. Consequently, the existence of useful information across complementing problems, opens doors for fruitful knowledge transfer. In practical settings, leveraging on such a rich pool of knowledge, can yield substantial cost-saving benefits [8, 9]. Thus, the ability to solve many tasks, with the scope of knowledge exchange among them, has been a longstanding goal of artificial intelligence systems targeting improved problem-solving efficacy [10].

Over the years, a handful of success stories have surfaced in the relatively young field of *transfer optimization* (TO), encompassing sequential transfers [11–16] as well as multitasking [17–24]. Similar to transfer learning, sequential transfer optimization utilizes high quality solutions from various source problems to solve a target optimization task. Therein, the transfer of knowledge (from past optimization problems to present) is conducted in a *unidirectional* fashion. That is, only a single target task is optimized at a time; re-optimization of the source task(s) is irrelevant. Solving multiple tasks concurrently is thus, beyond the scope of sequential transfer optimization techniques.

Multitasking effectively fills this void. Surprisingly, in spite of the longevity of *multitask learning*, the idea has only recently been introduced into the arena of *optimization problem-solving* [17, 18]. To elaborate, *multitask optimization* aims to facilitate efficient resolution of multiple problems simultaneously, by promoting *omnidirectional* knowledge transfer for greater synergistic search. Simply put, all the optimization tasks can in principle mutually help each other.

Despite its practical implications, preliminary works have indicated that the performance of multitask optimization is sensitive to the degree of underlying inter-task similarities. To elaborate, each optimization task may not always be closely related to all other available tasks. In such scenarios, assuming relatedness and solving (any) unrelated optimization problems concurrently can potentially lead to performance

slowdowns [25]. Overlooking such inter-task discrepancies and sharing knowledge across multiple diverse problems give rise to a phenomenon known as *negative transfer*. In fact, the threat of negative transfer has commonly been reported in the machine learning literature [6, 26], thereby highlighting a central pitfall of multitasking in general (i.e., be it in learning or in optimization).

Based on the above facts, in this paper, we propose a novel *data-driven* multitasking approach adept at *online* learning of *how much* knowledge to transfer across distinct optimization problems in multitask settings - without the need for any human intervention. We focus on the use of evolutionary algorithms (EAs) since they generally allow flexible data representations and have shown to combine well with probabilistic modeling approaches [27] - which forms the crux of our learning strategy. We first describe an instantiation of a multitasking evolutionary optimization framework from a probabilistic modeling perspective. Thereafter, we unveil how the principled integration of probabilistic models drawn from different tasks can actualize knowledge transfers while curbing negative inter-task interactions. It is noteworthy that the ideas described herein directly encompass a previously proposed *multifactorial evolutionary algorithm* (MFEA) for multitasking [18]. In comparison to the MFEA, our main contributions in this work are twofold:

- While the *mode* of transfer remains the same as MFEA (namely, crossover-based genetic transfer), the parameter inducing the *extent* of transfers is modified to take the form of a symmetric matrix. This allows effective multitasking across more than two tasks with possibly diverse inter-task relationships.
- The transfer parameter matrix is continuously learned and adapted *online* based on the data generated during the course of the multitasking search. As the result of a theoretically principled learning paradigm, the need for extensive manual parameter tuning is avoided.

With this, the novel algorithm put forward in this paper is labeled as MFEA-II. *We re-emphasize that in MFEA-II the mode of communication between tasks is primarily through crossover-based implicit genetic transfer.* It is worth noting that conceptually similar modes of transfer are often sought in distinct domains such as genetic programming and deep learning, where learned features (in the form of genetic programming sub-trees, or multiple layers of feature detectors) are reused across models trained on related datasets [5, 28, 29]. Importantly, this fast and scalable transfer mechanism is found to be effective in general black-box optimization settings as well, where we only have access to a finite set of solution-fitness pairs (without knowing their underlying relationship). In such settings, while constructing more complex transformations between tasks may be considered (see [17]), the associated learning algorithm consumes computational resources beyond typical applications of evolutionary methods.

The organization of the rest of this paper is as follows. Section II briefly discusses the backgrounds of the most notable approaches for multitask optimization, spanning Bayesian and evolutionary methods. An overview of mixture models

and subsequent theoretical developments in the context of multitasking are contained in Section III. In Section IV, we present an instantiation of a candidate evolutionary algorithm for multitask optimization, namely the MFEA. The theoretical foundations of the MFEA and its associated shortcomings are analyzed. Thereafter, the proposed online parameter estimation strategy for MFEA-II is presented in Section V. Experimental results and analyses on a series of discrete and continuous toy/benchmark examples are provided in Section VI. A practical case study on black-box neuroevolution-based controller design is carried out in Section VI-D. In particular, this section provides insights into the behavior of MFEA-II while tackling *many* tasks at once; with each task being a distinct variant of the popular double-pole balancing problem. Section VII concludes the paper with a brief discussion of future work.

## II. BACKGROUND AND RELATED WORK

In this section, we present a brief review of recent strides in the field of multitask optimization. The purpose of this section is to compare and contrast the outlined contributions of this paper to those available in the literature.

Multitask Bayesian optimization has been a prominent advancement in the context of bolstering hyperparameter optimization for automatic machine learning [17]. Its prime success is achieved by utilizing multitask Gaussian process models [26] to adaptively learn inter-task dependencies and improve the overall search of a Bayesian optimization algorithm. However, Bayesian optimization is generally limited to problems with fairly low or moderate dimensions. This is because, as the dimensionality of the search space increases, the number of data points required to ensure a good coverage of the space (for learning accurate Gaussian process models) increases exponentially (often referred to as the cold start problem) [30]. Furthermore, the applications of Bayesian optimization are primarily limited to continuous search spaces and do not directly apply to combinatorial search spaces wherein indefinite kernels may be difficult to deal with [31].

In contrast, evolutionary algorithms (EAs) have been promising in bridging these gaps. EAs offer immense flexibility in accommodating diverse data representations (be it continuous or combinatorial [32]), and scale fairly well to higher dimensions [33, 34]. Recently, evolutionary multitasking, under the label of multifactorial optimization [18], has demonstrated various success stories in solving multiple optimization problems simultaneously, encompassing discrete, continuous, single-, as well as multi-objective optimization [18, 19, 35–37]. Beyond this, evolutionary multitasking has also found its way in the field of machine learning; from training an ensemble of decision trees [38] to feedforward neural networks with modular knowledge representation [39].

Despite significant progress, it must be noted that existing evolutionary multitasking engines largely rely on prior assumption of the existence of exploitable synergies between tasks. In the absence of any knowledge about inter-task relationships, susceptibility to negative transfers have been highlighted [25, 40, 41]. Since the field of evolutionary multitasking is still in its infancy, efforts to identify inter-task

similarities and discrepancies *online* are currently under way. With this in mind, this paper presents a novel evolutionary multitasking approach for data-driven online learning of the latent similarities across problems; in turn promoting fruitful knowledge exchange between them. Our method generalizes well to a range of optimization problem types, as shall be showcased in the experimental study.

### III. OVERVIEW OF MIXTURE MODELS IN THE CONTEXT OF MULTITASK OPTIMIZATION

Herein, we present an overview of constructing and sampling probabilistic mixture distributions—that combine search distributions drawn from different tasks—as a way of initializing knowledge exchange in a multitask setting. It must be noted that for such a combination to be possible, all distributions must be defined in a common/shared space, which we refer to as the *unified search space*.

#### A. Unified search space

The importance of search space unification in multitasking is to encompass all the individual search spaces of different optimization tasks such that we have a shared platform on which the transfer of knowledge can seamlessly take place [7, 19, 42]. For the sake of brevity, consider  $K$  maximization tasks  $\{T_1, T_2, \dots, T_K\}$  to be solved concurrently. The search space dimensionality of each of the optimization tasks is given by  $D_1, D_2, \dots, D_K$  respectively. In such a scenario, we can define a *unified space*  $\mathbf{X}$  of dimensionality  $D_{unified} = \max\{D_1, D_2, \dots, D_K\}$ , such that there is sufficient flexibility to encompass each task in the multitask setting. The motivation behind using such a unification scheme, instead of simply concatenating the  $K$  distinct search spaces such that  $D_{unified} = (D_1 + D_2 + \dots + D_K)$ , is two-fold:

- Firstly, it helps circumvent the challenges associated with the curse of dimensionality when solving several tasks with multidimensional search spaces simultaneously.
- Secondly, it is considered to be an effective means of accessing the power of population-based search, promoting the discovery and implicit transfer of useful genetic materials from one task to another in an efficient manner.

With this, we propose  $\mathbf{X}$  to be limited to the range  $[0, 1]^{D_{unified}}$ , which serves as a *continuized* unified space into which all candidate solutions are mapped (*encoded*). Accordingly, while tackling the  $k^{th}$  task, a subset of  $D_k$  variables are extracted from a candidate solution  $x \in \mathbf{X}$  and *decoded* (inverse of the mapping function) to a task-specific solution representation for evaluation. For the case of box-constrained continuous optimization problems, a simple linear scaling offers a suitable one-to-one mapping between the task-specific solution space and the unified space. On the other hand, for the variety of discrete/combinatorial problems, different encoding/decoding processes can be conceived - readers are referred to [42] for a comprehensive overview.

#### B. Knowledge sharing via mixture distributions

Without loss of generality, for all  $k \in \{1, 2, \dots, K\}$ , let the (sub)populations associated with the  $k^{th}$  task be denoted as

$P^k$ . The corresponding objective function is denoted as  $f_k$  and  $f_k^* = f_k(x^*)$  denotes the global maximum of the  $k^{th}$  task. Note that we assumed  $x$  to be a candidate solution in the *unified search space*  $\mathbf{X}$ . With this background, a key assumption in the subsequent theoretical deductions of multitasking is that for any task  $T_k$ , the Borel measure of a set  $H = \{x|x \in \mathbf{X}, f_k(x) > f_k'\}$  is positive for any  $f_k' < f_k^*$ . In an evolutionary multitasking algorithm, let the subpopulations associated with each of the  $K$  tasks at time step  $t > 0$  originate from the underlying probability distributions  $p^1(x, t), p^2(x, t), \dots, p^K(x, t)$  respectively. Accordingly, in the process of facilitating *inter-task* interactions, the offspring population generated for the  $k^{th}$  task at the  $t^{th}$  iteration is considered to be drawn from the following mixture distribution,

$$q^k(x, t) = \alpha_k \cdot p^k(x, t) + \sum_{j \neq k} \alpha_j \cdot p^j(x, t). \quad (1)$$

The finite mixture  $q^k(x, t)$  is a linear combination of all  $K$  available distributions, with mixture coefficients given by the  $\alpha's$  [43]. Note that  $\alpha_k + \sum_{j \neq k} \alpha_j = 1$ , and  $\alpha's \geq 0$ .

#### C. Actualizing knowledge transfers in multitasking

The onset of knowledge transfer in multitasking occurs *during subsequent sampling of candidate solutions from the mixture distribution specified in Eq. (1)* [8, 44]. The degree of mixing (i.e., the amount of knowledge exchange) is mandated by the mixture coefficients  $\alpha's$ . It is noteworthy that in the absence of prior knowledge about the inter-task relationships, any inappropriate (blind) mixing of probability distributions will pose a threat of predominantly negative transfers by wrongly biasing the search distribution. To alleviate any such issues, it is imperative to facilitate the *optimal mixture* of the component densities by ensuring that the mixture coefficients ( $\alpha's$ ) are appropriately learned *online*.

#### D. Does Multitasking Prevent Global Convergence?

In this section, we highlight the effects of the mixture distribution on the global convergence behavior of multitasking. In particular, we show that asymptotic global convergence properties of probabilistic modeling based EAs continue to be preserved. For mathematical convenience, the theoretical study is carried out for large (sub)population sizes ( $N$ ), i.e., under the assumption that  $N \rightarrow \infty$ . Although such an assumption may not be completely practical, it is considered a reasonable simplification for developing the foundations of our proposals. Subsequently, the practical implications of our derived theoretical results are verified through experimental studies in Section VI.

In the limit of large values of  $N$ , it follows from the Glivenko-Cantelli theorem [45] that the empirical probability density of the (sub)population  $P^k$  closely approximates the true underlying distribution  $p^k(x)$ . Further, we assume that for the initial population,  $p^k(x, t=0)$  is positive and continuous in  $\mathbf{X}$  for all  $k \in \{1, 2, \dots, K\}$ . With this background, asymptotic convergence to the global optimum(s) of each constitutive

task  $k$  in a multitasking environment is said to occur if the following relation is satisfied,

$$\lim_{t \rightarrow \infty} \mathbb{E}[f_k(x)] = \lim_{t \rightarrow \infty} \int_{\mathbf{X}} f_k(x) \cdot p^k(x, t) \cdot dx = f_k^*, \forall k. \quad (2)$$

Loosely speaking, as time goes on in the optimization process, the distribution of the population must be gradually concentrated on only those points of the unified space that correspond to a global optimum.

For our theoretical deductions, we specifically consider the evolutionary multitasking algorithm to follow a  $(\mu, \lambda)$  selection strategy whereby  $\mu$  denotes the number of parents, and  $\lambda$  represents the number of offspring. Under this selection mechanism, the top  $\mu$  individuals out of  $\lambda$  offspring (with  $\mu < \lambda$ ), are selected for each task and taken as parents for the next iteration [46]. Note that the parent population of the  $k^{\text{th}}$  task is denoted as  $P^k$ . Further, the offspring population, of size  $\lambda = N$ , is denoted as  $P_c^k$ . Recall that the distribution of  $P_c^k$  is given by  $q^k(x)$  as in Eq. (1). For now, no elitism is assumed to strengthen the implications of our result - as randomized global search algorithms with elitism trivially possess asymptotic convergence properties.

To begin, let us define a parameter  $\theta = \mu/\lambda$ , such that  $0 < \theta < 1$ . Using the mixture distribution in Eq. (1), the  $(\mu, \lambda)$  parent selection scheme can be modeled as [47],

$$p^k(x, t+1) = \begin{cases} \frac{q^k(x, t)}{\theta} & \text{if } f_k(x) \geq \beta^k(t+1) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where  $\beta^k(t+1)$  represents a real number that satisfies,

$$\int_{f_k(x) \geq \beta^k(t+1)} q^k(x, t) \cdot dx = \theta. \quad (4)$$

Thus, the transition from  $p^k(x, t)$  to  $p^k(x, t+1)$  is based on Eqs. (1), (3) and (4).

**Theorem 1.** *Given  $p^k(x, 0)$  is positive and continuous in  $\mathbf{X}$  for all  $k$ , and  $N \rightarrow \infty$ , the asymptotic convergence to the global optimum(s) of each constitutive task is guaranteed during multitasking if  $\alpha_k > \theta$  in Eq. (1).*

*Proof.* Our deductions follow a similar style as [47]. At any iteration  $t > 0$ , the  $\mu$  parent samples associated with task  $k$  map to a set of objective values corresponding to  $f_k$ . Let this set be denoted as  $F^k(t)$ , and the infimum of the set as per Eq. (3) is  $\beta^k(t)$ , i.e.,  $\beta^k(t) = \inf F^k(t)$ . In other words, we have,

$$p^k(x, t) = 0 \Leftrightarrow f_k(x) < \beta^k(t). \quad (5)$$

This implies that a point is selected if and only if its objective function is not less than  $\beta^k(t)$ . We shall first show that  $\beta^k(t) < \beta^k(t+1)$ . This condition provides the intuition that the *spread of the population must be contracting*, gradually zooming in on the most promising regions of the unified space.

Based on Eq. (4), while selecting parents  $P^k(t)$  from offspring  $P_c^k(t-1)$ ,  $\beta^k(t)$  satisfies,

$$\int_{f_k(x) \geq \beta^k(t)} q^k(x, t-1) \cdot dx = \theta. \quad (6)$$

Notice that if the following inequality holds,

$$\int_{f_k(x) \geq \beta^k(t)} q^k(x, t) \cdot dx > \int_{f_k(x) \geq \beta^k(t+1)} q^k(x, t) \cdot dx, \quad (7)$$

then  $\beta^k(t) < \beta^k(t+1)$ . Invoking Eq. (1) for the left hand side of Eq. (7), we have,

$$\int_{f_k(x) \geq \beta^k(t)} q^k(x, t) \cdot dx \geq \int_{f_k(x) \geq \beta^k(t)} \alpha_k \cdot p^k(x, t) \cdot dx. \quad (8)$$

Moreover, Eq. (5) implies  $\int_{f_k(x) \geq \beta^k(t)} p^k(x, t) \cdot dx = 1$ , which on combining with Eq. (8) leads to the relation,

$$\int_{f_k(x) \geq \beta^k(t)} q^k(x, t) \cdot dx \geq \alpha_k. \quad (9)$$

Combining Eqs. (4), (7) and (9), we conclude that  $\beta^k(t) < \beta^k(t+1)$  holds if  $\alpha_k > \theta$ . The relation  $\beta^k(t) < \beta^k(t+1)$ , and the fact that  $\beta^k$  can not be greater than  $f_k^*$  implies that there exists a limit  $\lim_{t \rightarrow \infty} \beta^k(t) = f_k^*$  such that  $f_k^* \leq f_k^*$ . Hereafter, we present a proof that  $f_k^* = f_k^*$  via contradiction by first assuming,

$$f_k^* < f_k^*. \quad (10)$$

Based on the proposed transition procedure from  $p^k(x, t)$  to  $p^k(x, t+1)$  and per Eqs. (1) and (3), note,

$$p^k(x, t) \geq p^k(x, 0) \left[ \frac{\alpha_k}{\theta} \right]^t \quad \forall x : f_k(x) > f_k^*. \quad (11)$$

Let  $H = \{x | x \in \mathbf{X}, f_k(x) > f_k^*\}$ . Since  $p^k(x, 0) > 0$  for any  $x \in \mathbf{X}$ , and considering  $\frac{\alpha_k}{\theta} > 1$ , we have,

$$\lim_{t \rightarrow \infty} p^k(x, t) = +\infty, \forall x \in H. \quad (12)$$

Noting that the Borel measure of  $H$  is positive, and applying Fatou's lemma [48] to get,

$$\lim_{t \rightarrow \infty} \int_H p^k(x, t) \cdot dx = +\infty, \quad (13)$$

it is clear that Eq. (13) contradicts the fact that  $p^k(x, t)$  is a probability density function. Thus, the contradiction implies the conclusion,

$$\lim_{t \rightarrow \infty} \beta^k(t) = f_k^*. \quad (14)$$

□

#### IV. THE BASIC MULTIFACTORIAL EVOLUTIONARY ALGORITHM

In this section, we present an overview of an instantiation of a crossover and mutation based EA for multitask optimization. Recently proposed, the so-called multifactorial evolutionary algorithm (MFEA) [18] leverages on the implicit parallelism of population-based search to solve multiple problems simultaneously with *omnidirectional* transfer of knowledge across them. Under the relatively strong restriction of *parent-centric* evolutionary operators, crossover and mutation based EAs can be thought of as having resemblance to stochastic sampling-based optimization algorithms [27]. To elaborate, parent-centric operators tend to bias the offspring to be created nearer to the parents with high probability [49]. In the literature, there exists a family of operators that satisfy this assumption in

the context of real-coded EAs. Common examples include the simulated binary crossover (SBX) [50], polynomial mutation (PM) [51] and Gaussian mutation with small variance [52], to name a few. In such settings, it is not unreasonable (as shown below in Section IV-A) to consider that the empirical density of the offspring population  $P_c(t)$  tends to closely approximate that of the parent population  $P(t)$ ; which can be expressed as  $p_c(x, t) \approx p(x, t)$ . While such an assumption will not necessarily hold under arbitrary evolutionary operators, it provides sufficient simplification for an intuitive theoretical analysis that builds on the result of Section III. The practical implications of the same shall be demonstrated in the experimental sections.

### A. Preservation of parent distribution under parent-centric evolutionary operators

For simplicity, consider the probability distribution of a parent population to follow a multivariate Gaussian with mean  $m$  and covariance  $\Sigma$ . Additionally, let the mean and covariance of the offspring population be  $m_c$  and  $\Sigma_c$  respectively. It is noteworthy that parent-centric crossover operators already ensure  $m_c = m$  [53]. In general, we may write  $p_c(x, t) = p(x, t) + \delta_m$ , where  $\delta_m$  is a random vector with zero mean and covariance  $\delta\Sigma$ . Restricting the evolutionary operators to treat each variable independently (through variable-wise crossovers and/or mutations), we may write  $\Sigma_c = \Sigma + \delta\Sigma$ , where  $\delta\Sigma$  is a diagonal matrix of the form  $\delta\Sigma_{(i,i)} = \sigma_i^2$ . Here  $\sigma_i^2$  is the induced variance of the  $i^{\text{th}}$  variable during offspring creation. The specified high probability of creating offspring near the parents implies that  $\sigma_i^2 \ll \Sigma_{(i,i)}$ . Therefore,  $\Sigma_c \approx \Sigma$ , which suggests that the underlying distribution of the parent and offspring populations remain largely unaltered under variable-wise parent-centric operations.

### B. Preliminaries of the MFEA

The MFEA employs a single population  $P$  of individuals to solve  $K$  optimization tasks  $\{T_1, T_2, \dots, T_K\}$  simultaneously, where each task is seen as an added *factor* influencing the evolution of the population. The subpopulation associated with the  $k^{\text{th}}$  task is denoted as  $P^k$ . Given this background, certain terminologies commonly encountered in the context of the MFEA are as follows.

- **Definition 1 (Skill Factor):** The skill factor  $\tau_i$  of the  $i^{\text{th}}$  individual is the one task, amongst all  $K$  tasks in the multitasking environment, with which the individual is associated.
- **Definition 2 (Scalar Fitness):** The scalar fitness of the  $i^{\text{th}}$  individual in a multitasking environment is given by  $\varphi_i = 1/r_{\tau_i}^i$ , where  $r_{\tau_i}^i$  is the rank of the  $i^{\text{th}}$  individual on task  $\tau_i$ .

1) *Overview of the MFEA:* Algorithm 1 summarizes the general work flow of the present-day MFEA. The procedure evolves a population of  $N \cdot K$  individuals encoded in a unified space  $\mathbf{X}$ . Specifically, each of the  $K$  tasks is allocated equal resource of  $N$  individuals in its subpopulation. During the evolutionary process, in addition to the standard *intra-task crossover* between parents of the same skill factor, *inter-task crossover* can also occur between candidate solutions

associated with distinct optimization tasks, i.e., possessing different skill factors. It is this simple feature that opens the door for knowledge transfers through crossover-based genetic exchanges across tasks.

---

#### Algorithm 1: Pseudocode of a basic MFEA

---

```

1 Randomly sample  $N \cdot K$  individuals in  $\mathbf{X}$  to form initial population  $P(0)$ ;
2 for every individual  $p_i$  in  $P(0)$  do
3   Assign skill factor  $\tau_i = \text{mod}(i, K) + 1$ , for the case of  $K$  tasks;
4   Evaluate  $p_i$  for task  $\tau_i$  only;
5 Set  $t = 1$ ;
6 while stopping conditions are not satisfied do
7    $P(t) \leftarrow$  Select top  $N \cdot K/2$  ( $\leq 50\%$ ) of generated individuals (based on
   scalar fitness  $\varphi$ );
8   Configure offspring population  $P_c(t) = \emptyset$ ;
9   while offspring generated for each task  $< N$  do
10    Sample two individuals uniformly at random (without replacement):  $x_i$ 
    and  $x_j$  from  $P(t)$ ;
11    if  $\tau_i == \tau_j$  then
12       $[x_a, x_b] \leftarrow$  Intra-task crossover between  $x_i$  and  $x_j$ ;
13      Assign offspring  $x_a$  and  $x_b$  skill factor  $\tau_i$ ;
14    else if  $\text{rand} \leq \text{rmp}$  then
15       $[x_a, x_b] \leftarrow$  Inter-task crossover between  $x_i$  and  $x_j$ ;
16      Each offspring is randomly assigned skill factor  $\tau_i$  or  $\tau_j$ ;
17    else
18       $[x_a] \leftarrow$  local variation (mutation) of  $x_i$ ;
19      Assign offspring  $x_a$  skill factor  $\tau_i$ ;
20       $[x_b] \leftarrow$  local variation (mutation) of  $x_j$ ;
21      Assign offspring  $x_b$  skill factor  $\tau_j$ ;
22    Evaluate  $[x_a, x_b]$  for their assigned skill factors only;
23     $P_c(t) = P_c(t) \cup [x_a, x_b]$ ;
24   $t = t + 1$ ;

```

---

The extent of knowledge transfer in the MFEA is governed by a scalar (user-defined) parameter labeled as the *random mating probability (rmp)*. To elaborate on the role of the *rmp*, we refer to Step 14 in Algorithm 1. If two individuals selected from the parent pool correspond to different tasks, they can undergo crossover under a probability specified by  $\text{rmp} \in [0, 1]$ . During inter-task crossover, the genetic material created for one optimization task gets copied into the solution chromosome of an individual associated with a different task. As a result, implicit transfer of information is actualized across tasks. Clearly, if *rmp* is large, the transfer is more frequent than otherwise.

### C. Theoretical Foundations of the MFEA

At any generation  $t$  of the MFEA, the probability density function  $p(x, t)$  of the parent population  $P(t)$  is essentially a mixture of  $K$  probability densities  $p^k(x, t)$  representing the  $K$  tasks in the multitask setting. Herein, the  $k^{\text{th}}$  mixture component corresponds to subpopulation  $P^k(t) \subset P(t)$ . Importantly, while the *overall* offspring population distribution may satisfy  $p_c(x, t) \approx p(x, t)$  under appropriate assumptions on the evolutionary operators, the inter-task crossovers indicate that distribution  $p_c^k(x, t)$  need not necessarily be equivalent to  $p^k(x, t)$ . In particular, we have the following result on the mixture distribution occurring in the MFEA.

**Lemma 2.** *Under the stated assumptions of parent-centric evolutionary operators, the offspring mixture distribution  $p_c^k(x, t)$  of the  $k^{\text{th}}$  task in the MFEA can be expressed as,*

$$p_c^k(x, t) = \left[ 1 - \frac{0.5 \cdot (K-1) \cdot rmp}{K} \right] \cdot p^k(x, t) + \sum_{j \neq k} \frac{0.5 \cdot rmp}{K} \cdot p^j(x, t). \quad (15)$$

*Proof.* Refer to supplemental material.  $\square$

Note the resemblance between Eq. (15) and Eq. (1), with  $p_c^k(x, t)$  being analogous to  $q^k(x, t)$ .

1) *Does the MFEA Converge?:* Herein, we prove that the MFEA asymptotically converges to the global optimum(s) of each constitutive task regardless of the choice of  $rmp$ .

**Theorem 3.** *Under the large population assumption (i.e.,  $N \rightarrow \infty$ ), the MFEA with parent-centric evolutionary operators and  $(\mu, \lambda)$  selection possesses asymptotic global convergence properties across all tasks, given  $\theta (= \mu/\lambda) \leq 0.5$ .*

*Proof.* According to Theorem 1, for asymptotic global convergence of the  $k^{th}$  task in the multitasking environment, it suffices to ensure that  $\alpha_k > \theta$ . With reference to Lemma 2, the mixture coefficient  $\alpha_k$  corresponds to,

$$\alpha_k = 1 - \frac{0.5 \cdot (K-1) \cdot rmp}{K}. \quad (16)$$

Since  $0 \leq rmp \leq 1$ , it follows that  $0.5 < \alpha_k \leq 1$ , and thus  $\alpha_k$  is indeed greater than  $\theta$ . The above holds for all  $k$ .  $\square$

#### D. Susceptibility to Negative Inter-Task Interactions in Evolutionary Multitasking

As is clear from Algorithm 1, the performance of MFEA relies on the choice of  $rmp$ . Given the lack of prior knowledge about the inter-task relationships, tuning an appropriate  $rmp$  (via trial and error) thus becomes burdensome. Essentially, poor prescriptions of  $rmp$  risk the possibility of either predominantly negative inter-task interactions, or the potential loss of fruitful knowledge exchange.

In this section, we study the possible deleterious effects of negative inter-task interactions in evolutionary multitasking. In particular, we analyze the *rate of convergence* of the MFEA under the adverse case of no complementarity between constitutive tasks in a multitasking environment. A direct approach in this regard is to measure the evolution of the *spread* of subpopulations from generation  $t$  to  $t+1$  as  $\beta^k(t+1) - \beta^k(t)$ . Clearly, a large difference indicates a faster rate of convergence.

1) *Analyzing the Rate of Convergence of the MFEA:* Without loss of generality, we consider only the  $k^{th}$  optimization task as the same arguments hold across all  $K$  tasks. Given  $N \rightarrow \infty$ , we simulate the adverse scenario of no complementarity between constitutive tasks, at some generation  $t (> 0)$  of the MFEA, by assuming that wherever  $p^k(x, t) > 0$ , we have  $p^j(x, t) = 0$ , for all  $j \neq k$ . By extension, the following also holds,

$$f_k(x) \geq \beta^k(t) \Rightarrow p^j(x, t) = 0, \text{ for all } j \neq k. \quad (17)$$

Qualitatively speaking, the optimum solution(s) of the  $k^{th}$  task do not lie in the neighborhood of the optimum solution(s) of any other task being optimized in the same multitasking environment. With this, the effects of negative interactions

between optimization tasks can be analyzed by comparing the expected progress (from generation  $t$  to  $t+1$ ) of a traditional single-tasking solver (denoted by  $ST$ ) against its multitasking counterpart. We make the simplification that the population distribution at the  $t^{th}$  generation is identical for both solvers; i.e.,  $p^{k,ST}(x, t) \approx p^k(x, t)$  and  $\beta^{k,ST}(t) \approx \beta^k(t)$ .

Notably, due to the absence of inter-task crossovers in single-tasking, the offspring distribution  $p_c^{k,ST}(x, t)$  is *not* a mixture model (unlike MFEA). Further, Section IV-A shows that *parent-centric* operators result in  $p_c^{k,ST}(x, t) \approx p^k(x, t)$ . Thus, for the  $(\mu, \lambda)$  selection with  $\theta \leq 0.50$ ,  $\beta^{k,ST}(t+1)$  satisfies,

$$\int_{f_k(x) \geq \beta^{k,ST}(t+1)} p_c^{k,ST}(x, t) \cdot dx \approx \int_{f_k(x) \geq \beta^{k,ST}(t+1)} p^k(x, t) \cdot dx = \theta (= 0.5 - \epsilon), \text{ where } \epsilon \in [0, 0.5]. \quad (18)$$

On the other hand, since  $\beta^k(t+1) > \beta^k(t)$  (Theorem 1), Eq. (15) together with Eq. (17) imply that  $\beta^k(t+1)$  must satisfy,

$$\int_{f_k(x) \geq \beta^k(t+1)} p_c^k(x, t) \cdot dx = \int_{f_k(x) \geq \beta^k(t+1)} \left[ 1 - \frac{0.5 \cdot (K-1) \cdot rmp}{K} \right] \cdot p^k(x, t) \cdot dx = 0.5 - \epsilon. \quad (19)$$

For *strictly positive* values of  $rmp$ , we have the inequality  $0.5 < \left[ 1 - \frac{0.5 \cdot (K-1) \cdot rmp}{K} \right] < 1$ . Then, combining Eqs. (18) and (19) leads to,

$$\int_{f_k(x) \geq \beta^k(t+1)} p^k(x, t) \cdot dx = [0.5 - \epsilon] \cdot \left[ 1 - \frac{0.5 \cdot (K-1) \cdot rmp}{K} \right]^{-1} > \int_{f_k(x) \geq \beta^{k,ST}(t+1)} p^k(x, t) \cdot dx, \quad (20)$$

which indicates  $\beta^{k,ST}(t+1) > \beta^k(t+1)$ . Therefore,

$$\beta^{k,ST}(t+1) - \beta^{k,ST}(t) > \beta^k(t+1) - \beta^k(t). \quad (21)$$

Eq. (21) suggests that in the case where there is no complementarity between tasks, single-tasking may lead to faster convergence characteristics than multitasking. Further, the result of Eq. (20) shows that the reduction in the convergence rate of MFEA depends on the chosen  $rmp$ .

#### V. TRANSFER PARAMETER ESTIMATION VIA OPTIMAL MIXTURE MODELING

In this section, we propose an online  $rmp$  estimation technique that theoretically guarantees to minimize the negative (harmful) interactions between distinct optimization tasks.

From the convergence rate analysis of MFEA we infer that negative transfers in multitasking can be suppressed by forcing  $p_c^k(x, t)$  to replicate  $p^k(x, t)$  as closely as possible. Based on Lemma 2, this can easily be accomplished by assigning  $rmp = 0$ . However, such a naive strategy prohibits inter-task crossovers and, as a result, prohibits any knowledge transfer from occurring (see Algorithm 1). With this in mind, our primary goal is to put forward a strategy that promotes knowledge transfer ( $rmp > 0$ ), while simultaneously mitigating negative inter-task interactions. To this end, we propose a novel data-driven approach for online learning of  $rmp$  values that lead to an optimal mixture distribution in Eq. (15).

It is worth mentioning that hereafter,  $RMP$  is no longer a scalar parameter, but takes the form of a symmetric  $K \times$

$K$  matrix (given  $K$  optimization tasks) of pairwise values, represented as,

$$RMP = \begin{bmatrix} rmp_{1,1} & rmp_{1,2} & \dots \\ rmp_{2,1} & rmp_{2,2} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} \quad (22)$$

where  $rmp_{j,k} = rmp_{k,j}$  captures the synergy (or the extent of transfer) between the  $j^{th}$  and  $k^{th}$  tasks. Accordingly, it is naturally enforced that  $rmp_{j,j} = 1, \forall j$ . *This enhancement offers the distinct advantage that in many practical scenarios, where the complementarity between tasks may not be uniform across different task-pairs, our proposed method shall be capable of capturing non-uniform inter-task synergies.*

At generation  $t$ , consider  $g^k(x, t)$  to be a probabilistic model (estimate) of the true distribution  $p^k(x, t)$  of any  $k^{th}$  task of interest. Herein,  $g^k(x, t)$  is built from the dataset of subpopulation  $P^k(t)$ . By replacing the true density functions with the learned probabilistic models, and using the elements of the  $RMP$  matrix instead of the scalar  $rmp$ , Eq. (15) can be rewritten as,

$$g_c^k(x, t) = \left[ 1 - \frac{0.5}{K} \cdot \sum_{k \neq j} rmp_{k,j} \right] \cdot g^k(x, t) + \frac{0.5}{K} \sum_{j \neq k} rmp_{k,j} \cdot g^j(x, t). \quad (23)$$

wherein  $g_c^k(x, t)$  is a probabilistic mixture model approximating the distribution of the offspring population:  $p_c^k(x, t)$ .

Thus, in order to force  $p_c^k(x, t)$  to closely replicate  $p^k(x, t)$ , it is equivalent to learning the  $RMP$  matrix such that the offspring probabilistic model  $g_c^k(x, t)$  accurately models the parent distribution  $p^k(x, t)$ , for all  $k \in \{1, 2, \dots, K\}$ .

#### A. Learning the RMP Matrix Online

The first step is to build probabilistic models  $g^k(x, t)$  on each subpopulation  $P^k(t)$ , for all  $k \in \{1, 2, \dots, K\}$ . From here onwards, consider the parameter  $\theta (= \mu/\lambda)$  to be 0.5. Then, given the  $N/2$  parent solutions in each subpopulation  $P^k(t)$  at some iteration  $t$ , we propose to learn the  $RMP$  matrix by maximizing the following log-likelihood function:

$$\max_{RMP} \sum_{k=1}^K \sum_{i=1}^{N/2} \log g_c^k(x_{ik}, t). \quad (24)$$

where  $x_{ik}$  is the  $i^{th}$  sample (individual) in the dataset of  $P^k(t)$ .

We claim that the data-driven learning paradigm of Eq. (24) serves to reduce the ‘‘gap’’ between  $g_c^k(x, t)$  and  $p^k(x, t)$ , measured over all  $k$ . In this regard, we note that a commonly encountered quantifier of distribution gaps is the Kullback Leibler ( $KL$ ) divergence [54]. Specifically, the  $KL$  divergence provides the amount of information lost when an arbitrary density function  $g$  is used to approximate another density  $p$ . The mathematical equation corresponding to this definition is,

$$KL(p||g) = \int_{\mathbf{X}} p(x) \cdot [\log p(x) - \log g(x)] \cdot dx. \quad (25)$$

In what follows, we prove our claim in the context of the  $KL$  divergence.

**Theorem 4.** *Maximizing the likelihood function in Eq. (24) is equivalent to minimizing the gap between the probability densities  $g_c^k(x, t)$  and  $p^k(x, t)$  averaged across all tasks  $\{T_1, T_2, \dots, T_k, \dots, T_K\}$ .*

*Proof.*  $\sum_{i=1}^{N/2} \log g_c^k(x_{ik}, t)$  can also be written as,

$$(N/2) \frac{\sum_{i=1}^{N/2} \log g_c^k(x_{ik}, t)}{N/2} = \frac{N}{2} \mathbb{E} [\log g_c^k(x_{ik}, t)]. \quad (26)$$

where  $\mathbb{E}[\cdot]$  is the expected value. Under the assumption of large  $N$ , the empirical probability density function induced by the samples in  $P^k(t)$  converges to the underlying probability density  $p^k(x, t)$  [45]. Thus, we have,

$$\mathbb{E} [\log g_c^k(x, t)] = \int_{\mathbf{X}} p^k(x, t) \cdot \log g_c^k(x, t) \cdot dx. \quad (27)$$

Accordingly, we can restate the mathematical formulation of Eq. (24) as,

$$\max_{RMP} \sum_{k=1}^K \int_{\mathbf{X}} p^k(x, t) \cdot \log g_c^k(x, t) \cdot dx. \quad (28)$$

Note that maximizing Eq. (28) is equivalent to the following minimization problem, given  $p^k(x, t)$  to be fixed,

$$\min_{RMP} \sum_{k=1}^K \int_{\mathbf{X}} p^k(x, t) \cdot [\log p^k(x, t) - \log g_c^k(x, t)] \cdot dx. \quad (29)$$

Referring to Eq. (25), we can express Eq. (29) in terms of the  $KL$  divergence as:

$$\min_{RMP} \sum_{k=1}^K KL(p^k(x, t) || g_c^k(x, t)). \quad (30)$$

In other words, the mathematical program of Eq. (24) indeed minimizes the gap (given by the  $KL$  divergence) between the mixture models  $g_c^k(x, t)$  and the true density functions  $p^k(x, t)$ , averaged across all  $K$  optimization tasks.  $\square$

Theorem 4 highlights that through the proposed data-driven learning of the  $RMP$  matrix, the various tasks in a multitask setting can be directly brought to bear upon one another, while facilitating the suppression of negative transfers. In other words, the method results in a theoretically principled adaptive online transfer parameter estimation scheme.

With regard to the online learning paradigm, an important matter of ‘‘practical’’ interest is the choice of probabilistic model type for building  $g^k(x, t)$ , for all  $P^k(t) \in \{P^1(t), P^2(t), \dots, P^K(t)\}$ . To this end, observe that choosing a complex probabilistic model that may tend to overfit the (in practice reasonably small) dataset  $P^k(t)$ , potentially leads to the cancellation of inter-task transfers. This is in fact a well documented phenomenon even in the classical transfer/multitask learning literature, which is typically overcome by internal cross-validation steps or noise incorporation for preventing overfitting [26, 55]. Along similar lines, in this paper we propose to use simple and fast probabilistic modeling approaches (e.g., univariate marginal distributions) to help prevent the overfitting issue; which in turn allows the various other models in the multitask setting to be mobilized for filling in distribution gaps.

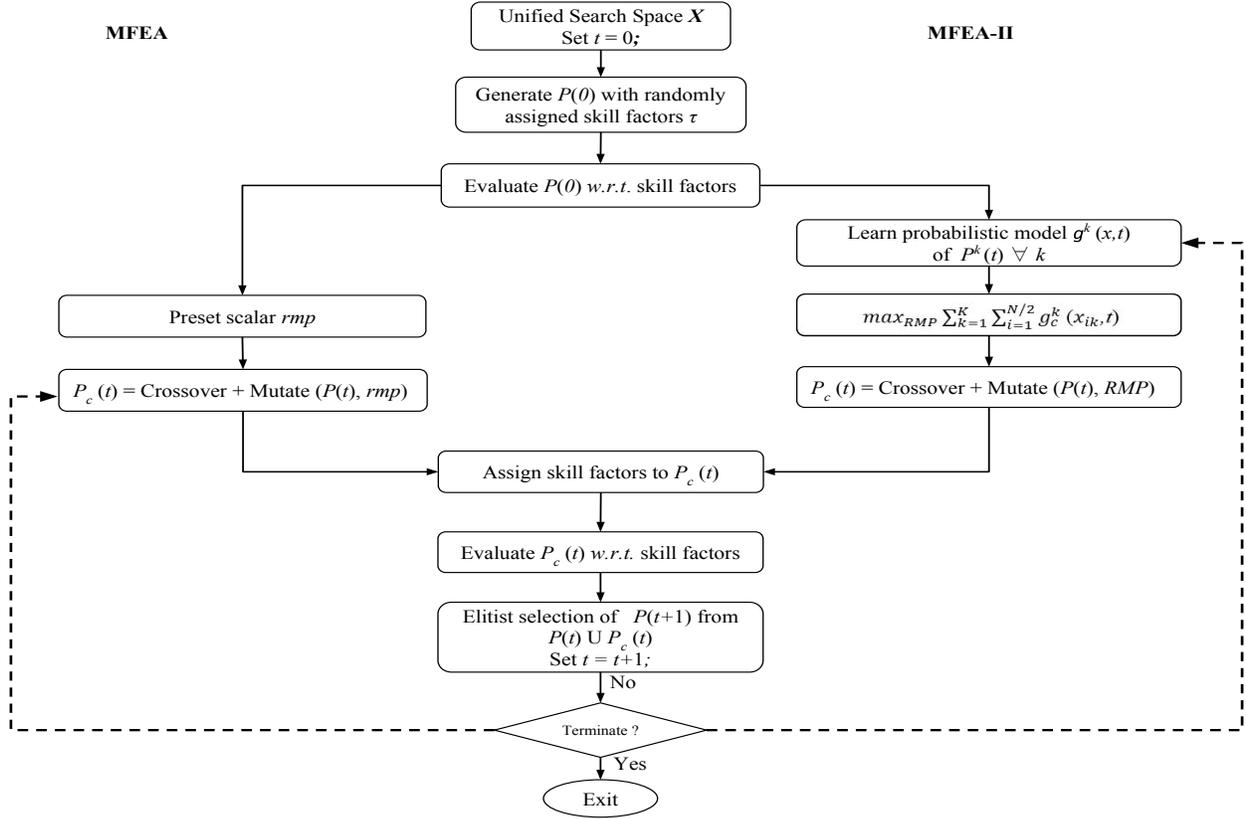


Fig. 1. **Left:** General framework of the existing MFEA. Notice that MFEA employs an *offline* scalar  $rmp$  assignment. **Right:** MFEA-II with the added *online*  $RMP$  matrix learning module.

### B. Main Ingredients of MFEA-II

This section introduces the enhanced MFEA-II incorporated with the proposed online  $RMP$  matrix learning scheme. Algorithm 2 provides the pseudocode of the  $RMP$  learning procedure (Section V-A). The general steps associated with MFEA-II are illustrated by Figure 1 (right side). As depicted, the initial phases of MFEA-II are basically similar to the existing MFEA. The distinguishing facet of the MFEA-II is the incorporation of the online  $RMP$  learning module within the overall optimization algorithm. It is important to highlight that given built probabilistic models  $g^k(x, t)$ , the mathematical problem of Eq. (24) is *convex upward*. As a result, it is found to be solved to optimality at little computational overhead using classical optimizers.

---

#### Algorithm 2: Online $RMP$ learning

---

- 1 Build simple and fast models  $g^k(x, t)$  on each subpopulation  $P^k(t)$ ,  $\forall k \in \{1, 2, \dots, K\}$ .
  - 2 Optimize mixture models:  $\max_{RMP} \sum_{k=1}^K \sum_{i=1}^{N/2} \log g_c^k(x_{ik}, t)$ .
- 

To summarize, in MFEA-II, the offspring population  $P_c(t)$  is generated based on the learned  $RMP$  matrix, which automatically ascertains the extent of genetic transfer across distinct tasks. Note that the main modification compared to the original MFEA occurs during inter-task crossover in steps 14 – 21 of Algorithm 1. The modified steps of MFEA-II are summarized in Algorithm 3, wherein solutions corresponding

to different tasks undergo crossover based on the learned  $RMP$  matrix. Further, note that during inter-task crossovers, a parent-centric operator (without additional uniform crossover-like variable swap) is applied so as to ensure that this key assumption of our theoretical deduction does indeed hold in the present instantiation of MFEA-II. Each generated offspring also undergoes small parent-centric mutation. It is worthwhile to mention that if the learned  $RMP$  matrix is full of 0's, the MFEA-II works similar to multiple canonical EAs executing single-tasking in parallel. The ability of the MFEA-II to identify discrepancies and interpret similarities across multiple tasks on the fly is exemplified through subsequent experimental studies.

---

#### Algorithm 3: Inter-task crossover in MFEA-II

---

- 1 Randomly sample two individuals  $x_i$  and  $x_j$  from  $P(t)$ ;
  - 2 **if**  $\tau_i \neq \tau_j$  **then**
  - 3     **if**  $\text{rand} \leq rmp_{\tau_i, \tau_j}$  **then**
  - 4          $[x_a, x_b] \leftarrow$  *Inter-task crossover* between  $x_i$  and  $x_j$  ;
  - 5         Each offspring is randomly assigned skill factor  $\tau_i$  or  $\tau_j$  ;
  - 6     **else**
  - 7         Randomly select  $x'_i$  with skill factor  $\tau_i$  ;
  - 8          $[x_a] \leftarrow$  *Intra-task crossover* between  $x_i$  and  $x'_i$  ;
  - 9         Assign offspring  $x_a$  skill factor  $\tau_i$  ;
  - 10         Randomly select  $x'_j$  with skill factor  $\tau_j$  ;
  - 11          $[x_b] \leftarrow$  *Intra-task crossover* between  $x_j$  and  $x'_j$  ;
  - 12         Assign offspring  $x_b$  skill factor  $\tau_j$  ;
-

## VI. EXPERIMENTAL STUDY

In this section, numerical illustrations are presented that showcase the efficacy of the proposed online parameter estimation scheme of the MFEA-II. Herein, we conduct experimental studies on a variety of problems, ranging from discrete to continuous benchmark optimization. In addition, we investigate the practicality of our proposition on increasingly challenging variants of the double-pole balancing controller design task [56]. For rigorous analysis, the performance of the MFEA-II is compared against a number of baseline solvers. Firstly, we consider the basic counterparts, i.e., the present day multifactorial evolutionary algorithm (denoted as MFEA-I) [18] and the standalone canonical EA (CEA). Further, for the case of continuous double-pole balancing problem, two state-of-the-art variants of *natural evolutionary strategies* (NES) [57] are considered as additional baselines for comparison. These include the *separable natural evolutionary strategy* (SNES) [58] and the *exponential natural evolutionary strategy* (xNES) [59]. It is worthwhile to mention that NES has garnered much recent attention as a powerful evolutionary solver, with notable applications in OpenAI<sup>1</sup> projects on reinforcement learning [60].

### A. Experimental Configuration

The experimental setup is outlined in what follows. The population size  $N$  (per task) is kept the same for all algorithms, ensuring consistency. Specifically, if a single-tasking algorithm utilizes a population size  $N$  for each of the  $K$  tasks, then a multitasking algorithm tackling the same  $K$  tasks is assigned a population size of  $N \cdot K$ . The general algorithmic settings for all the problems are:

- 1) Continuized unified representation with range  $[0, 1]^{D_{unified}}$ ;
- 2) Evolutionary operators for MFEA-II, MFEA-I and CEA:
  - a) SBX crossover with probability ( $p_c$ ) = 1 and distribution index  $\eta_c = 15$ ;
  - b) Polynomial mutation with probability ( $p_m$ ) =  $1/d$  and distribution index  $\eta_m = 15$ ;
- 3) Probabilistic model for MFEA-II: variable-wise (marginal) normal distribution;
- 4) For the pedagogical case study:
  - a) Population size ( $N$ ): 100;
  - b) Maximum function evaluations: 15,000;
- 5) For the synthetic continuous benchmark set:
  - a) Population size ( $N$ ): 50
  - b) Maximum function evaluations: 500,000;
- 6) For the double-pole balancing problem:
  - a) Population size ( $N$ ): 100;
  - b) Maximum function evaluations: 10,000;
- 7) *rmpr*: Learned online for MFEA-II. For MFEA-I:
  - a) 0.3 for the pedagogical study and the synthetic continuous benchmarks (as per [25]);
  - b) 0.1 for the double-pole balancing (tuned for best performance).

Note that we utilize the SBX and polynomial mutation operators since they agree with the underlying assumptions of the theoretical derivations in Section IV.

The implementations of the NES variants are obtained from [61]. Based on [62], the optimized set of parameters are found to be:

- 1) For SNES, the learning rates for the search parameters mean and standard deviation are  $\eta_\mu = 1$  and  $\eta_\sigma = 0.1058$ , respectively.
- 2) For xNES, the learning rates for the mean and covariance components are  $\eta_\mu = 1$  and  $\eta_\sigma = \eta_B = 0.0017$ .

We note that the aforementioned optimized NES parameters are close to default settings. Both NES variants incorporate rank based fitness shaping, as was suggested in [57].

### B. Pedagogical Case Study

We begin by considering a set of binary optimization problems as toy examples. While decoding candidate solutions from the continuous unified space ( $\mathbf{X} = [0, 1]^{D_{unified}}$ ) to the discrete/binary space, a variable is assigned a value of 0 if its encoded counterpart is less than or equal to  $\leq 0.5$ . Else, it is assigned a value of 1.

For this particular study, three popular binary problems from literature are studied, namely, *onemax*, *zeromax* and the highly deceptive *trap-5* problem [63]. While the optimum of *onemax* is the string of all 1's, that of *zeromax* is the string of all 0's. In *trap-5*, the string is first partitioned into consecutive groups of 5 non-overlapping bits. Further, a 5-bit trap function is applied to each of the groups, and the contribution of each group towards the combined objective function is given as follows.

$$trap_5(u) = \begin{cases} 5 & \text{if } u = 5 \\ (4 - u) & \text{if } u < 5 \end{cases} \quad (31)$$

where  $u$  is the number of 1's in the group. Observe there exists only one global optimum of the *trap-5* problem, when all the bits in the  $D$ -dimensional input string are 1's, and  $(2^{\frac{D}{5}} - 1)$  other local optima.

The rationale behind the choice of the above set of problems is that they provide an ideal platform to showcase the potential advantages of the proposed online parameter estimation scheme of MFEA-II. To elaborate, the optimum solution of *onemax* is seen to intersect with the global optimum of *trap-5*, implying that the crossover-based exchange of genetic material across tasks is indeed a suitable mode of transfer. However, due to the presence of a large number of local optimum in *trap-5*, it is deemed that the uninhibited inter-task crossovers between the two tasks is unfavorable, since this can potentially hinder the convergence characteristics of *onemax*. Additionally, the *zeromax* problem is clearly conflicting with *onemax* in terms of solution similarity, and by extension, with the *trap-5* problem as well. Nonetheless, the worst (highly deceptive) local optimum of *trap-5* corresponds to the global optimum of *zeromax*. Therefore, it is contended that solving the three tasks concurrently using the existing MFEA-I (with manually specified scalar *rmpr*), would be susceptible to negative transfers.

<sup>1</sup><https://openai.com/>

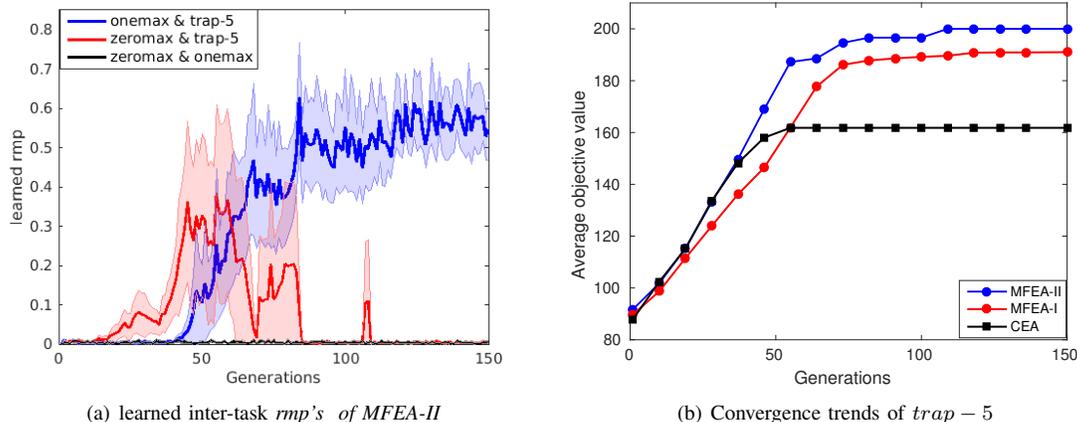


Fig. 2. **Left:** Pairwise  $rmp$ 's learned between distinct tasks over successive generations. The shaded regions span one half standard deviation on either side of the mean. **Right:** Convergence trends (averaged over 30 independent runs) achieved by all the algorithms on the trap-5 problem. As shown, both multitasking variants are superior to CEA, with MFEA-II being the most dominant.

In this study, we solve dimensionality  $D = 200$  variants of each of the three problems. The multitasking variants MFEA-I and MFEA-II tackle all the three tasks simultaneously, while CEA solves each in isolation. A summary of the experimental results averaged over 30 independent runs are contained in Figure 2. Note that a  $3 \times 3$  symmetrical  $RMP$  matrix is learned for MFEA-II, capturing the pairwise inter-task similarities over successive generations. The learned  $RMP$  matrix elements are presented in Figure 2(a).

Figure 2(b) shows the convergence trends of the highly deceptive trap-5 problem. For brevity, the convergence trends of one-max and zero-max problems are omitted. In general, it can be observed that the scope for genetic transfer in both multitasking variants (MFEA-I and MFEA-II) has been beneficial, while the single-tasking CEA gets constantly trapped in the deceptive local optimum. Further, upon a closer look at Figure 2(b), it can be seen that the convergence behavior of MFEA-I has been fairly inferior to that of MFEA-II. Notice that MFEA-I does not always converge to the global optimum. This is primarily due to the uninhabited exchange of genetic materials in the MFEA-I, making it vulnerable to negative transfers. However, this is clearly not the case with MFEA-II, which exploits the relationships learned from task datasets to optimally mandate knowledge transfers (at different stages of the optimization phase). As is depicted by Figure 2(a), during the initial stages, when the transfer across the three tasks is unlikely to be helpful, the learned  $RMP$  matrix values are indeed low. Progressively, once high quality genetic material is available for transfer from onemax to the trap-5 problem (around after 50 generations), the online parameter estimation scheme starts prescribing higher values of the corresponding inter-task  $rmp$ . As indicated by the high  $rmp$  values, the positive transfers from onemax guide the trap-5 to consistently (and quickly) converge to the global optimum. We also note that relatively low to medium  $rmp$  values are learned between zeromax and trap-5, implying that the search is to some extent drawn towards the highly deceptive local optimum of trap-5 (under the influence of zeromax). In contrast, negligible

(near zero)  $rmp$  values are learned between conflicting problem pairs onemax and zeromax throughout the evolutionary search. As a result, the MFEA-II successfully identifies and exploits positive transfer (when available), while circumventing the negative exchanges plaguing MFEA-I; see Figure 2(b).

For further analysis of the performance of MFEA-II on other toy examples, the reader is referred to the supplemental material.

### C. Synthetic continuous benchmark functions

In the next set of examples, we showcase the  $rmp$  learned given *known* inter-task relationships between task pairs. To this end, we utilize some of the CEC composite minimization benchmark functions comprising of high, medium and low inter-task similarities [25]. The inter-task similarity measure

TABLE I  
BENCHMARK PROBLEM SETS [25]

Category	Task	$D$	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 NI+HS	Rosenbrock ( $T_1$ )	50	multimodal, nonseparable	0.9434
	Rastrigin ( $T_2$ )	50	multimodal, nonseparable	
5 NI+MS	Griewank ( $T_1$ )	50	multimodal, nonseparable	0.3669
	Weierstrass ( $T_2$ )	50	multimodal, nonseparable	

between a pair of tasks ( $R_s$ ) is based on the Spearman rank correlation between their respective function landscapes. Accordingly, the problem pairs with  $R_s < 0.2$  are considered to have low similarity (LS), problem pairs with  $0.2 \leq R_s < 0.8$  are defined as medium similarity (MS), and pairs with  $R_s \geq 0.8$  imply high inter-task similarity (HS). Further, the global optima of each of the task-pairs may *completely intersect* in the unified search space, denoted as CI, or may *not intersect*, denoted as NI. The dimensions of each of the tasks is represented by  $D$  in Table I.

To begin with, we analyze Table II containing the numerical results achieved by MFEA-II, MFEA-I and CEA on the composite benchmark problems. A task is assumed to be solved if it reaches an objective function value within a convergence tolerance threshold of  $1e - 04$ , hence recorded as 0.00.

TABLE II

COMPARISON OF MFEA-II WITH MFEA-I AND CEA ON THE CONTINUOUS MINIMIZATION BENCHMARK PROBLEM SETS. THE MEAN AND STANDARD DEVIATIONS OF 30 INDEPENDENT RUNS ARE PROVIDED. BOLDFACE ENTRIES MARKED WITH \* ARE BEST AND SIGNIFICANTLY THE BEST (BY WILCOXON TEST WITH SIGNIFICANCE LEVEL 0.05). BEST BUT NOT SIGNIFICANTLY THE BEST ENTRIES ARE ONLY BOLDFACE.

Problem set	MFEA-II		MFEA-I		CEA	
	$T_1$	$T_2$	$T_1$	$T_2$	$T_1$	$T_2$
1 CI+HS	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	82.35
	0.00	0.00	0.00	0.00	0.00	11.51
2 CI+MS	<b>0.00</b>	<b>0.00*</b>	<b>0.00</b>	5.57	18.82	82.17
	0.00	0.00	0.00	7.26	0.92	12.17
3 CI+LS	<b>19.98</b>	<b>1.67E+03*</b>	20.00	3.60e+03	20.01	1.94E+03
	0.04	5.18E+02	0.002	7.26E+02	0.01	4.34E+02
4 NI+HS	<b>55.42</b>	<b>4.08*</b>	58.51	13.10	65.99	83.89
	29.91	9.73	47.99	34.33	32.52	13.06
5 NI+MS	<b>0.00</b>	<b>20.61</b>	<b>0.00</b>	21.08	<b>0.00</b>	31.52
	0.00	1.19	0.00	1.72	0.00	4.96

It is seen that MFEA-II demonstrates consistently superior performance overall. Not only has MFEA-II performed generally better in the cases of high similarities, it has also outperformed MFEA-I and CEA for the problem pair with low inter-task similarity (CI+LS). In particular, it can be observed that MFEA-II has improved the performance of MFEA-I on  $T_2$  of CI+LS. This is because the lack of inter-task similarities makes the MFEA-I susceptible to predominantly negative transfer of genetic materials. On the contrary, MFEA-II dynamically modulates the extent of genetic transfers in order to minimize such harmful negative inter-task interactions. Most importantly, the same performance benefits continues to hold for the no intersection (NI) cases as well.

In order to demonstrate the efficacy of the online *RMP* learning module, we analyze the learned pairwise *rmp* curves of MFEA-II across various task-pairs characterized by different degrees of similarities. Figure 3 reveals that the MFEA-II successfully deciphers the underlying inter-task similarities *online*. Notice from Figures 3(a) and 3(b) that high values of *rmp* are learned (close to 1) for problem sets CI+HS and NI+HS that were known to have high inter-task similarities ( $R_s \approx 1$ ). Similarly, relatively lower *rmp* values are learned for the problem pairs NI+MS and CI+LS with lower inter-task similarity; see Figures 3(c) and 3(d). Intuitively, based on the above observations, we conjecture that the online *rmp* estimation scheme indeed captures synergies (and identifies differences) across diverse optimization tasks in multitask settings.

To give a concrete example of the added benefits of online parameter estimation, we show the convergence behavior of MFEA-II against its counterparts MFEA-I and CEA for the CI+HS case. As is clear from Figure 4, MFEA-II is overall superior to MFEA-I and CEA. It can be seen that MFEA-II converges significantly faster (within 100,000 FE's) in

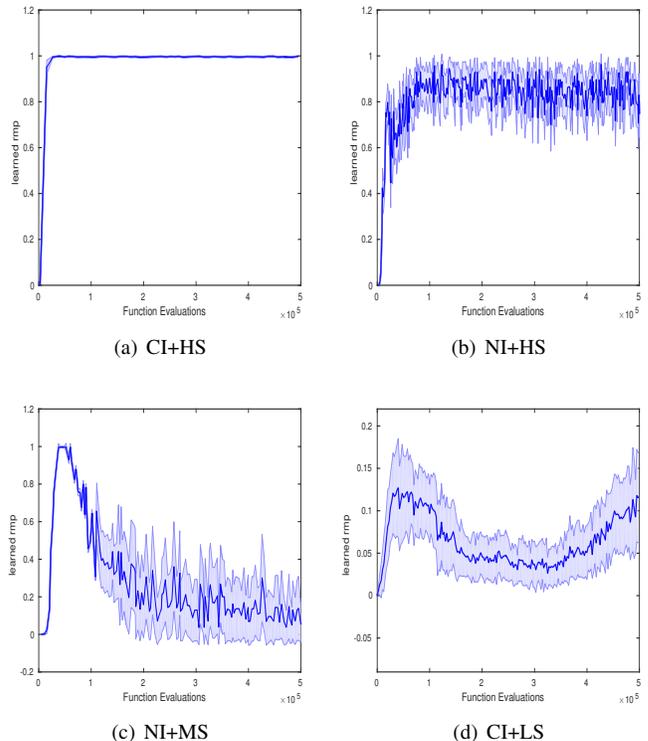


Fig. 3. Learned pairwise *rmp* values between pairs of tasks achieved by MFEA-II on different problem sets characterized by different degrees of inter-task similarities. The plots are averaged over 30 independent runs with the shaded regions spanning one-half standard deviation on either side of the mean.

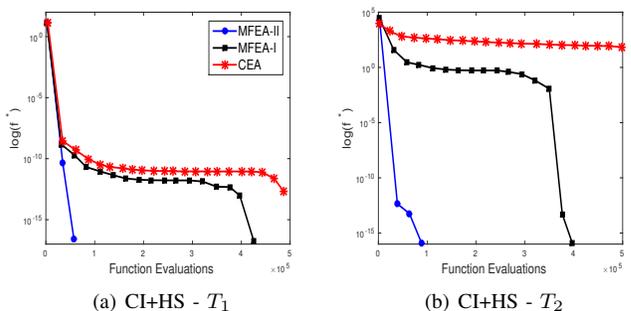


Fig. 4. Convergence trends (averaged over 30 independent runs) achieved by the three tested algorithms on the problem set CI+HS with high degrees of inter-task similarity.

comparison to MFEA-I on both tasks,  $T_1$  and  $T_2$ . The reason behind this accelerated convergence of MFEA-II can be inferred from the learned *rmp* trend in Figure 3(a).

#### D. A Practical Case Study of Multitasking

In previous works, the applicability of multitasking in diverse real-world domains, including operations research [64], engineering [19], machine learning [65], etc., has already been demonstrated. Accordingly, in this section, our goal is more towards analyzing the behavior of the newly proposed MFEA-II in practical settings. We consider a neuroevolution-based approach to simultaneously solving multiple variants of the double-pole balancing controller design task. In particular, we highlight that in a multitasking environment, not only can

complex tasks benefit from simpler ones, but the reverse can also hold (as a consequence of omnidirectional knowledge transfer). Notably, this is a feature unique to multitasking that is not available in sequential transfer procedures. We further showcase the efficacy of the MFEA-II in the face of *many* tasks occurring at once.

In literature, the double-pole balancing problem (or inverted pendulum system) has been a standard benchmark for artificial learning systems [56, 66–69]. The problem involves balancing two poles (of different lengths) hinged on a cart, using a neural network controller, such that the poles do not fall beyond a predefined vertical angle and the cart remains within the bounds of a horizontal track. Typically, the length of the longer pole remains fixed to  $1.0m$  while the length of the shorter pole  $l_s$  maybe varied. As substantiated in literature [56], the task of balancing both the poles become progressively harder as the poles assume similar lengths. In other words, the system becomes difficult to control as the length of the shorter pole  $l_s$  approaches to  $1.0m$ .

As such, it gets challenging (even for state-of-the-art optimizers) to solve such difficult control tasks from scratch (zero-state knowledge). Accelerating the acquisition of grounded knowledge by learning with many goals (tasks) has been a promising approach in the domain of reinforcement learning systems. It has been shown that while intentionally attempting to solve the harder tasks, other simpler tasks can be implicitly (unintentionally) solved [10]. *Furthermore, in this study we demonstrate that during multitasking, both simple and hard tasks can mutually benefit from one another as a unique consequence of the omnidirectional transfer of knowledge.*

In our experiments, we consider the Markovian case of the double-pole task. To elaborate, all state variables, including, the cart position-  $x$ , the cart velocity-  $\dot{x}$ , angle of longer pole-  $\theta_1$ , angular velocity of longer pole-  $\dot{\theta}_1$ , angle of shorter pole-  $\theta_2$ , and the angular velocity of shorter pole-  $\dot{\theta}_2$ , are available as inputs to the neural network. The associated equations of motions and experimental parameters are based on the descriptions provided in [68, 70].

The dynamical system is evaluated using the fourth-order Runge-Kutta numerical integration scheme with step size  $\tau = 0.01s$ . That is to say, this is a simulation-based black-box optimization problem since the analytical form of the objective function is unknown. A simple feedforward neural network (FNN) controller, with 10 hidden neurons, is used. The optimization problem is about automatically tuning the weights of the FNN controller. Accordingly, the network outputs a force between  $[-10, 10]N$  every 0.02 seconds. This force is then applied to the cart and the next state is updated, which is reintroduced as the new input to the neural network. The process continues until a failure occurs or the desired control action has been performed for the required amount of time. The amount of time before failure thus becomes the measurable fitness. The initial position of the longer pole is set to  $\theta_1 = 1^\circ$  and the shorter pole remains upright ( $\theta_2 = 0^\circ$ ). The track was set to  $4.8m$  long. For the task to be solved, the angles of the poles have to be in the specified range  $[-36^\circ, 36^\circ]$  for  $10^5$  time steps, where each step corresponds to  $0.02s$ . This is equivalent to over 30 minutes of simulated time.

### 1) Omnidirectional Transfer in Multitask Optimization:

In this particular study, we highlight the benefits of *omnidirectional* transfer in multitasking. As opposed to *sequential transfer* methods which tackle *one* task at a time through *unidirectional* knowledge transfers (from past to present), multitasking offers the flexibility of handling multiple tasks concurrently. With this in mind, we demonstrate that the omnidirectional genetic transfer regime of MFEA-II makes it possible for all tasks to gain maximum benefits from one another. To showcase this facet of multitasking, we refer to the numerical results contained in Table III. Therein, three tasks of different short pole lengths  $l_s$  (with increasing difficulty levels) are considered. The percentage of successful runs (out of 30 independent runs) is used as a performance measure to compare MFEA-II against CEA.

TABLE III  
PERCENTAGE SUCCESS RATE (OUT OF 30 RUNS) OF THREE DIFFERENT TASKS. THEY ARE EITHER SOLVED SEPARATELY (CEA), OR MERGED INTO PAIRS OR A TRIPLET (MFEA-II).

Task	$l_s$	CEA	MFEA-II			
			$(T_1, T_2)$	$(T_1, T_3)$	$(T_2, T_3)$	$(T_1, T_2, T_3)$
$T_1$	$0.60m$	27%	30%	30%	-	47%
$T_2$	$0.65m$	0%	27%	-	27%	37%
$T_3$	$0.70m$	0%	-	7%	27%	17%

According to the results, it can be seen that by tackling all the three tasks separately, the CEA can only solve task  $T_1$  (27%) and fails to find suitable control strategies for the more challenging variants  $T_2$  and  $T_3$ , within the allocated computational budget. respectively. On the other hand, the performance consistently improves if we tackle two distinct tasks simultaneously using MFEA-II. The dash symbol (-) in Table III implies the *absence* of a particular task in the MFEA-II. In addition to improving the success rate of task  $T_1$  (30%), multitasking also helps to improve the performance of harder instances  $T_2$  and  $T_3$  when compared to single tasking. Eventually, pooling all the three tasks together  $(T_1, T_2, T_3)$ , significantly improves the overall performance of tasks  $T_1$  (47%) and  $T_2$  (37%). This highlights the fact that genetic transfers from complex tasks such as  $T_3$  ( $l_s = 0.70m$ ), can also help to boost the performance of simpler tasks  $T_1$  and  $T_2$ .

2) *Some insights on Many Tasking:* We extend our experimental setup to a series of six different tasks of different short pole lengths  $l_s$ , ranging from  $0.45m$  to  $0.70m$ . Further, two state-of-the-art solvers SNES [58] and xNES [59] are included as additional baselines for comparison. The percentage success rate (out of 30 runs) of each of the algorithms, on each task is summarized in Table IV.

Overall, the numerical results reveal that the MFEA-II is superior to other methods, on almost all the tasks under consideration. While MFEA-I, SNES and xNES have demonstrated competitive performance on the first three tasks  $(T_1, T_2, T_3)$ , their performances rapidly drop as the length  $l_s$  increases. Apart from MFEA-II, the success rate of all other methods is significantly low on the increasingly challenging variants

TABLE IV  
COMPARISON OF VARIOUS METHODS ON DIFFERENT SHORT POLE LENGTHS  $l_s$ , FOR THE DOUBLE-POLE BALANCING PROBLEM. THE TABLE SHOWS THE PERCENTAGE SUCCESS RATE (%) OUT OF 30 INDEPENDENT RUNS. BEST ENTRIES ARE HIGHLIGHTED IN BOLD.

Task	$l_s$	Success Rate (%)			
		MFEA-II	MFEA-I	SNES [58]	xNES [59]
$T_1$	$0.45m$	<b>100%</b>	80%	<b>100%</b>	97%
$T_2$	$0.50m$	<b>100%</b>	80%	77%	97%
$T_3$	$0.55m$	97%	60%	73%	<b>100%</b>
$T_4$	$0.60m$	<b>83%</b>	50%	27%	43%
$T_5$	$0.65m$	<b>63%</b>	20%	13%	10%
$T_6$	$0.70m$	<b>37%</b>	10%	0%	0%

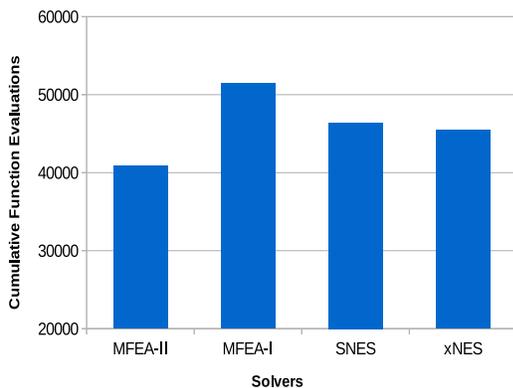


Fig. 5. Comparison of total number of function evaluations consumed by each of the algorithms in solving all the six different tasks.

$T_5$  ( $l_s = 0.65m$ ) and  $T_6$  ( $l_s = 0.70m$ ). MFEA-II has improved the success for the last two more difficult control tasks  $T_5$  and  $T_6$ , with 63% and 37% success respectively. On a closer look, it can be observed that MFEA-I has also performed better than the standalone solvers on the last two difficult task instances. In fact, SNES and xNES are found to be unsuccessful in effectively balancing both the poles for  $T_6$  with  $l_s = 0.70m$ . As an important aside, the total number of function evaluations required by MFEA-II to solve all the six different tasks is measurably less than the rest of the solvers, as shown by Figure 5.

Further, Figure 6 shows the pairwise  $rmp_{j,k}$  values learned between distinct task-pairs  $T_j$  and  $T_k$ , during the course of the optimization phase in MFEA-II. The curves highlight the existence of underlying inter-task synergies between 15 possible task-pairs of different short pole lengths  $l_s$ . Accordingly, the amount of transfers is regulated by the learned  $rmp$ 's in the  $RMP$  matrix, which are seen to vary across different stages of the optimization process. Recall that if  $rmp$  is high, the frequency of transfer is high. In contrast, the frequency of transfer in MFEA-I remains fixed by the preset value of  $rmp$ .

Importantly, Figure 6 highlights that MFEA-II is able to autonomously distinguish similar tasks from dissimilar ones. To elaborate, it is worth noting that tasks with short poles of similar lengths are intuitively expected to have more synergy.

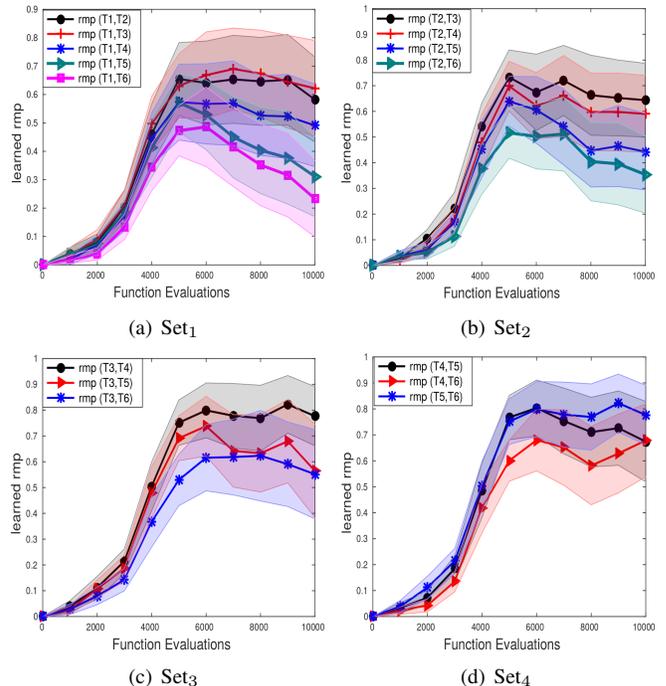


Fig. 6. Pairwise  $rmp$ 's of MFEA-II, learned between constitutive tasks of different short pole lengths. The plots are averaged over 30 independent runs and the shaded regions span one-half standard deviation on either side of the mean.

Accordingly, from Table IV, it can be said that  $T_1$  is more similar to  $T_2$  than it is to  $T_3/T_4/T_5/T_6$ . Similarly,  $T_2$  is more similar to  $T_3$  than it is to  $T_4/T_5/T_6$ . Figure 6 shows that MFEA-II precisely captures this trend. In other words, as the expected inter-task similarity decreases, so do the learned  $rmp$  values.

## VII. CONCLUSION

In this paper, we proposed a novel online transfer parameter estimation scheme for evolutionary multitasking, capable of learning and exploitation of inter-task synergies on the fly. As a first step to our proposition, we theoretically analyze the shortcomings of the existing evolutionary multitasking framework MFEA-I from the standpoint of its susceptibility to negative transfers. Accordingly, we introduce the MFEA-II, integrated with *online transfer parameter estimation* to dynamically control the extent of knowledge exchange across tasks. Specifically, the extent of transfer is adapted based on the optimal blending (mixing) of probabilistic models in order to capture inter-task similarities *online* in a purely data-driven manner. The algorithmic contributions of MFEA-II can be summarized from two perspectives:

- The transfer parameter takes the form of a symmetric matrix for effectively multitasking across more than two tasks with possibly diverse inter-task relationships.
- The transfer parameter matrix is learned and adapted *online* during the course of the multitasking search.

The practicality of the MFEA-II was demonstrated experimentally on a series of synthetic as well as practical optimization problems. Essentially, the experimental results reveal that the algorithm is indeed adept at exploiting the

similarities and discrepancies between tasks during multitask optimization, thereby allaying any fear of harmful inter-task interactions. Notably, in the context of complex controller design tasks, the utility of the online knowledge transfer scheme is clearly revealed. Furthermore, our proposed method is automatic, alleviates human intervention (for ascertaining inter-task synergies), and yields consistently superior performance in comparison to some state-of-the-art optimizers.

Looking into the future, emerging platforms such as cloud computing and the internet of things will offer large-scale data storage and seamless communication facilities, thereby making it possible for embedded optimization solvers to take advantage of the knowledge (data) made available by related tasks. In such settings, the general principles of multitasking are expected to play a key role in simultaneously solving multiple tasks (both more efficiently and effectively) by exploiting recurring real-world patterns. This notion of multitasking lends itself to a form of *General Optimization Intelligence* (GOI) - characterized by universal machines capable of performing varied tasks and learning with experience.

#### ACKNOWLEDGEMENT

This work is partially supported by the Singapore Institute of Manufacturing Technology-Nanyang Technological University (SIMTech-NTU) Joint Laboratory and Collaborative research Programme on Complex Systems.

#### REFERENCES

- [1] S. Thrun, "Is learning the n-th thing any easier than learning the first?" in *Advances in neural information processing systems*, 1996, pp. 640–646.
- [2] 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.
- [3] M. Iqbal, W. N. Browne, and M. Zhang, "Extracting and using building blocks of knowledge in learning classifier systems," in *Proceedings of the 14th annual conference on Genetic and evolutionary computation*. ACM, 2012, pp. 863–870.
- [4] —, "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.
- [5] M. Iqbal, B. Xue, H. Al-Sahaf, and M. Zhang, "Cross-domain reuse of extracted knowledge in genetic programming for image classification," *IEEE Transactions on Evolutionary Computation*, 2017.
- [6] R. Caruana, "Multitask learning," in *Learning to learn*. Springer, 1998, pp. 95–133.
- [7] 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.
- [8] A. Gupta, Y.-S. Ong, and L. Feng, "Insights on transfer optimization: Because experience is the best teacher," *IEEE Transactions on Emerging Topics in Computational Intelligence*, p. In Press.
- [9] A. Gupta and Y.-S. Ong, *Memetic Computation: The Main-spring of Knowledge Transfer in a Data-Driven Optimization Era*. Springer, 2019.
- [10] S. Cabi, S. G. Colmenarejo, M. W. Hoffman, M. Denil, Z. Wang, and N. de Freitas, "The intentional unintentional agent: Learning to solve many continuous control tasks simultaneously," *arXiv preprint arXiv:1707.03300*, 2017.
- [11] P. Cunningham and B. Smyth, "Case-based reasoning in scheduling: reusing solution components," *International Journal of Production Research*, vol. 35, no. 11, pp. 2947–2962, 1997.
- [12] M. Kaedi and N. Ghasem-Aghae, "Biasing bayesian optimization algorithm using case based reasoning," *Knowledge-Based Systems*, vol. 24, no. 8, pp. 1245–1253, 2011.
- [13] M. Pelikan, M. W. Hauschild, and P. L. Lanzi, "Transfer learning, soft distance-based bias, and the hierarchical boa," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2012, pp. 173–183.
- [14] L. Feng, Y.-S. Ong, A.-H. Tan, and I. W. Tsang, "Memes as building blocks: a case study on evolutionary optimization+ transfer learning for routing problems," *Memetic Computing*, vol. 7, no. 3, pp. 159–180, 2015.
- [15] M. Jiang, Z. Huang, L. Qiu, W. Huang, and G. G. Yen, "Transfer learning based dynamic multiobjective optimization algorithms," *arXiv preprint arXiv:1612.06093*, 2016.
- [16] B. Da, A. Gupta, and Y.-S. Ong, "Curbing negative influences online for seamless transfer evolutionary optimization," *IEEE Transactions on Cybernetics*, no. 99, pp. 1–14, 2018.
- [17] K. Swersky, J. Snoek, and R. P. Adams, "Multi-task bayesian optimization," in *Advances in neural information processing systems*, 2013, pp. 2004–2012.
- [18] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: towards evolutionary multitasking," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 343–357, 2016.
- [19] A. Gupta, Y.-S. Ong, L. Feng, and K. C. Tan, "Multiobjective multifactorial optimization in evolutionary multitasking," *IEEE Transactions on cybernetics*, 2017.
- [20] L. Feng, L. Zhou, J. Zhong, A. Gupta, Y.-S. Ong, K.-C. Tan, and A. Qin, "Evolutionary multitasking via explicit autoencoding," *IEEE transactions on cybernetics*, no. 99, pp. 1–14, 2018.
- [21] A. Gupta and Y.-S. Ong, "Back to the roots: Multi-x evolutionary computation," *Cognitive Computation*, pp. 1–17, 2019.
- [22] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multitasking for evolutionary optimization of expensive problems," *IEEE Transactions on Evolutionary Computation*, 2017.
- [23] J. Tang, Y. Chen, Z. Deng, Y. Xiang, and C. P. Joy, "A group-based approach to improve multifactorial evolutionary algorithm," in *IJCAI*, 2018, pp. 3870–3876.
- [24] R.-T. Liaw and C.-K. Ting, "Evolutionary manytasking optimization based on symbiosis in biocoenosis," in *Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.
- [25] 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>.
- [26] E. V. Bonilla, K. M. Chai, and C. Williams, "Multi-task gaussian process prediction," in *Advances in neural information processing systems*, 2008, pp. 153–160.
- [27] M. Pelikan, M. W. Hauschild, and F. G. Lobo, "Estimation of distribution algorithms," in *Springer Handbook of Computational Intelligence*. Springer, 2015, pp. 899–928.
- [28] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1717–1724.
- [29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in neural information processing systems*, 2014, pp. 3320–3328.
- [30] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.
- [31] M. Zaefferer and T. Bartz-Beielstein, "Efficient global optimization with indefinite kernels," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2016, pp. 69–79.
- [32] H. Mühlenbein, M. Gorges-Schleuter, and O. Krämer, "Evolution algorithms in combinatorial optimization," *Parallel Com-*

- puting, vol. 7, no. 1, pp. 65–85, 1988.
- [33] M. A. Potter and K. A. De Jong, “A cooperative coevolutionary approach to function optimization,” in *International Conference on Parallel Problem Solving from Nature*. Springer, 1994, pp. 249–257.
- [34] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, “Cooperative coevolution with differential grouping for large scale optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 378–393, 2014.
- [35] Q. Chen, X. Ma, Y. Sun, and Z. Zhu, “Adaptive memetic algorithm based evolutionary multi-tasking single-objective optimization,” in *Asia-Pacific Conference on Simulated Evolution and Learning*. Springer, 2017, pp. 462–472.
- [36] R.-T. Liaw and C.-K. Ting, “Evolutionary many-tasking based on biocoenosis through symbiosis: A framework and benchmark problems,” in *Evolutionary Computation (CEC), 2017 IEEE Congress on*. IEEE, 2017, pp. 2266–2273.
- [37] Y.-W. Wen and C.-K. Ting, “Parting ways and reallocating resources in evolutionary multitasking,” in *Evolutionary Computation (CEC), 2017 IEEE Congress on*. IEEE, 2017, pp. 2404–2411.
- [38] —, “Learning ensemble of decision trees through multifactorial genetic programming,” in *Evolutionary Computation (CEC), 2016 IEEE Congress on*. IEEE, 2016, pp. 5293–5300.
- [39] R. Chandra, A. Gupta, Y.-S. Ong, and C.-K. Goh, “Evolutionary multi-task learning for modular knowledge representation in neural networks,” *Neural Processing Letters*, pp. 1–17, 2017.
- [40] 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*, 2016.
- [41] K. K. Bali, A. Gupta, L. Feng, Y. S. Ong, and T. P. Siew, “Linearized domain adaptation in evolutionary multitasking,” in *Evolutionary Computation (CEC), 2017 IEEE Congress on*. IEEE, 2017, pp. 1295–1302.
- [42] A. Gupta, J. Mañdziuk, and Y.-S. Ong, “Evolutionary multitasking in bi-level optimization,” *Complex & Intelligent Systems*, vol. 1, no. 1-4, pp. 83–95, 2015.
- [43] P. Smyth and D. Wolpert, “Stacked density estimation,” in *Advances in neural information processing systems*, 1998, pp. 668–674.
- [44] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer Science & Business Media, 2001, vol. 2.
- [45] L. D. L. Gyorfi, G. Lugosi, and L. Devroye, “A probabilistic theory of pattern recognition,” 1996.
- [46] H.-G. Beyer, *The theory of evolution strategies*. Springer Science & Business Media, 2013.
- [47] Q. Zhang and H. Muhlenbein, “On the convergence of a class of estimation of distribution algorithms,” *IEEE Transactions on evolutionary computation*, vol. 8, no. 2, pp. 127–136, 2004.
- [48] Y. Chow and H. Yeicher, “Probability theory, 3rd ed.” 1997.
- [49] K. Deb, A. Anand, and D. Joshi, “A computationally efficient evolutionary algorithm for real-parameter optimization,” *Evolutionary computation*, vol. 10, no. 4, pp. 371–395, 2002.
- [50] R. B. Agrawal, K. Deb, and R. Agrawal, “Simulated binary crossover for continuous search space,” *Complex systems*, vol. 9, no. 2, pp. 115–148, 1995.
- [51] K. Deb and M. Goyal, “A combined genetic adaptive search (geneas) for engineering design,” *Computer Science and informatics*, vol. 26, pp. 30–45, 1996.
- [52] R. Hinterding, “Gaussian mutation and self-adaption for numeric genetic algorithms,” in *Evolutionary Computation, 1995., IEEE International Conference on*, vol. 1. IEEE, 1995, p. 384.
- [53] K. Deb, D. Joshi, and A. Anand, “Real-coded evolutionary algorithms with parent-centric recombination,” in *Evolutionary Computation, 2002. CEC’02. Proceedings of the 2002 Congress on*, vol. 1. IEEE, 2002, pp. 61–66.
- [54] S. Kullback and R. A. Leibler, “On information and sufficiency,” *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.
- [55] D. Pardoe and P. Stone, “Boosting for regression transfer,” in *Proceedings of the 27th International Conference on International Conference on Machine Learning*. Omnipress, 2010, pp. 863–870.
- [56] F. Gomez, J. Schmidhuber, and R. Miikkulainen, “Accelerated neural evolution through cooperatively coevolved synapses,” *Journal of Machine Learning Research*, vol. 9, no. May, pp. 937–965, 2008.
- [57] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, “Natural evolution strategies,” *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 949–980, 2014.
- [58] T. Schaul, T. Glasmachers, and J. Schmidhuber, “High dimensions and heavy tails for natural evolution strategies,” in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 2011, pp. 845–852.
- [59] T. Glasmachers, T. Schaul, S. Yi, D. Wierstra, and J. Schmidhuber, “Exponential natural evolution strategies,” in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. ACM, 2010, pp. 393–400.
- [60] T. Salimans, J. Ho, X. Chen, and I. Sutskever, “Evolution strategies as a scalable alternative to reinforcement learning,” *arXiv preprint arXiv:1703.03864*, 2017.
- [61] T. Schaul, “Natural evolution strategies,” Nov. 2017. [Online]. Available: <http://people.idsia.ch/~tom/nes.html>
- [62] M. López-Ibáñez, J. Dubois-Lacoste, T. Stützle, and M. Bittantari, “The irace package, iterated race for automatic algorithm configuration,” Technical Report TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium, Tech. Rep., 2011.
- [63] K. Sastry, D. E. Goldberg, and M. Pelikan, “Limits of scalability of multiobjective estimation of distribution algorithms,” in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, vol. 3. IEEE, 2005, pp. 2217–2224.
- [64] Y. Yuan, Y.-S. Ong, A. Gupta, P. S. Tan, and H. Xu, “Evolutionary multitasking in permutation-based combinatorial optimization problems: Realization with TSP, QAP, LOP, and JSP,” in *Region 10 Conference (TENCON), 2016 IEEE*. IEEE, 2016, pp. 3157–3164.
- [65] R. Chandra, A. Gupta, Y.-S. Ong, and C.-K. Goh, “Evolutionary multi-task learning for modular training of feedforward neural networks,” in *International Conference on Neural Information Processing*. Springer, 2016, pp. 37–46.
- [66] R. S. Sutton, “Temporal credit assignment in reinforcement learning,” 1984.
- [67] F. J. Gomez and R. Miikkulainen, “Solving non-markovian control tasks with neuroevolution,” in *IJCAI*, vol. 99, 1999, pp. 1356–1361.
- [68] K. O. Stanley and R. Miikkulainen, “Evolving neural networks through augmenting topologies,” *Evolutionary computation*, vol. 10, no. 2, pp. 99–127, 2002.
- [69] C. Igel, “Neuroevolution for reinforcement learning using evolution strategies,” in *Evolutionary Computation, 2003. CEC’03. The 2003 Congress on*, vol. 4. IEEE, 2003, pp. 2588–2595.
- [70] A. P. Wieland, “Evolving neural network controllers for unstable systems,” in *Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on*, vol. 2. IEEE, 1991, pp. 667–673.



**Kavitesh Kumar BALI** is currently a PhD student in the School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore. Bali received his MSc degree in Computer Science from the University of the South Pacific (USP), Fiji, in 2016. His current research interests are in evolutionary computation, transfer optimization and machine learning.



**Yew-Soon ONG** (M99-SM12-F18) received the Ph.D. degree in artificial intelligence in complex design from the University of Southampton, U.K., in 2003. He is currently a Professor at the School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore, where he is also Director of the Data Science and Artificial Intelligence Research Center and Singtel-NTU Cognitive & Artificial Intelligence Corporate Laboratory. His research interest in artificial intelligence spans across memetic computing, optimization intelligence, and

machine learning. Dr. Ong is the founding Editor-in-Chief of the IEEE Transactions on Emerging Topics in Computational Intelligence, Technical Editor-in-Chief of Memetic Computing and associate editor of the IEEE Transactions on Evolutionary Computation, the IEEE Transactions on Neural Networks & Learning Systems, the IEEE Transactions on Cybernetics, and others. He has received several IEEE outstanding paper awards and listed as a Thomson Reuters highly cited researcher and among the World's Most Influential Scientific Minds.



**Abhishek GUPTA** received his PhD in Engineering Science from the University of Auckland, New Zealand, in the year 2014. He graduated with a Bachelor of Technology degree in the year 2010, from the National Institute of Technology Rourkela, India. He currently serves as a Scientist in the Singapore Institute of Manufacturing Technology (SIMTech), Agency of Science, Technology and Research (A\*STAR), Singapore. He has diverse research experience in the field of computational science, ranging from numerical methods in engineering physics, to topics in computational intelligence. His recent research interests are in the development of memetic computation as an approach for automated knowledge extraction and transfer across problems in evolutionary design.

design.



**Puay Siew TAN** leads the Manufacturing Control Tower™ (MCT™) as the Programme Manager which is responsible for the setup of Model Factory@SIMTech. Her research has been in the cross-field disciplines of Computer Science and Operations Research for cyber physical production system (CPPS) collaboration, in particular sustainable complex manufacturing and supply chain operations. To this end, she has been active in using context-aware and services techniques. Her present research interests include Complex Systems, specifically quantitative techniques for understanding disruptions propagation in networked supply chains and mitigation of risks caused by these disruptions.

chains and mitigation of risks caused by these disruptions.

# Supplemental Material

## I. PROOF OF LEMMA 2

*Proof.* Let an offspring solution  $x_a$  be assigned a skill factor  $k$  i.e.,  $\tau_a = k$ . Given the scope of *parent-centric* inter-task crossover,  $x_a$  may have been drawn from either of the densities  $p^1(x, t), p^2(x, t), \dots, \dots, \text{or}, p^k(x, t), \dots, \text{or}, p^K(x, t)$  corresponding to  $K$  distinct tasks in the multitasking environment. By Bayes' theorem, the probability that  $x_a$  is drawn from  $p^k(x, t)$ , denoted as  $x_a \sim p^k(x, t)$  is given by the following:

$$P(x_a \sim p^k(x, t) | \tau_a = k) = \frac{P(\tau_a = k | x_a \sim p^k(x, t)) \cdot P(x_a \sim p^k(x, t))}{P(\tau_a = k)}. \quad (1)$$

Considering the uniform allocation of resources to each of the  $K$  optimization tasks in the MFEA, we have,

$$P(x_a \sim p^k(x, t)) = P(\tau_a = k) = \frac{1}{K}. \quad (2)$$

Hence, by further simplification of Eq. 1, we have,

$$P(x_a \sim p^k(x, t) | \tau_a = k) = P(\tau_a = k | x_a \sim p^k(x, t)). \quad (3)$$

The condition that  $x_a$  originates from  $p^k(x, t)$  implies that at least one of its parents must also belong to  $p^k(x, t)$ . In other words, at least one of its parent must have a skill factor  $k$ . According to Algorithm 1, the probabilistic event  $P(\tau_a = k | x_a \sim p^k(x, t))$  occurs under three possible scenarios given as follows.

Scenario-1: *intra-task crossover* occurs where the second randomly chosen parent of  $x_a$  also has skill factor  $k$ . Accordingly, the probability of the first scenario is given by,

$$P(\text{Scenario} - 1) = \frac{1}{K}. \quad (4)$$

Scenario-2: *inter-task crossover* with probability  $rpm$ , after which  $x_a$  is assigned skill factor  $k$  with probability 0.5. Thus, the probability of the second case is expressed as,

$$P(\text{Scenario} - 2) = \frac{0.5 \cdot (K - 1) \cdot rpm}{K}. \quad (5)$$

Scenario-3:  $x_a$  is created by a small (local) variation of the parent with skill factor  $k$ . The probability of the third case is given by,

$$P(\text{Scenario} - 3) = \frac{(K - 1) \cdot (1 - rpm)}{K}. \quad (6)$$

Consequently we have,

$$P(\tau_a = k | x_a \sim p^k(x, t)) = \sum_{i=1}^3 P(\text{Scenario} - i). \quad (7)$$

Thus,

$$P(x_a \sim p^k(x, t) | \tau_a = k) = 1 - \frac{0.5 \cdot (K - 1) \cdot rpm}{K}. \quad (8)$$

Similarly, it can be shown that,

$$P(x_a \sim p^j(x, t) \forall j \neq k | \tau_a = k) = \sum_{j \neq k} \frac{0.5 \cdot rpm}{K}. \quad (9)$$

Combining Eq. (8) and Eq (9), the overall mixture distribution  $p_c^k(x, t)$  can be expressed as,

$$p_c^k(x, t) = [1 - \frac{0.5 \cdot (K - 1) \cdot rpm}{K}] \cdot p^k(x, t) + \sum_{j \neq k} \frac{0.5 \cdot rpm}{K} \cdot p^j(x, t). \quad (10)$$

□

## II. VISUALIZATION OF SYNTHETIC BENCHMARKS

### A. Visualization of Complete Intersection (CI) Case in 1-D

For elaboration, the notion of complete intersection of global optima is illustrated in Figure 1. The figure represents a projection in the unified space, and shows that the global optima of the two tasks  $T_1$  and  $T_2$  are located at the same point in the unified search space.

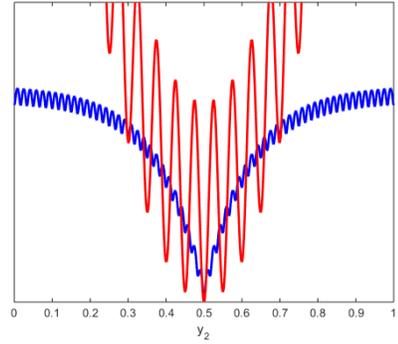


Fig. 1. In the complete intersection case, the global optimum of all tasks maps to the same point in the unified search space.

### B. Visualization of No Intersection (NI) Case in 1-D

The notion of no intersection of global optima is illustrated in Figure 2. From the figure, we see that the global optima of the two tasks are no longer at the same point, but are separated from each other, in the unified search space.

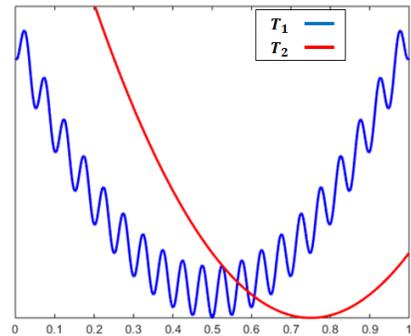


Fig. 2. In the no intersection case, the global optima of the two tasks  $T_1$  and  $T_2$  are separated.

### III. FURTHER EXPERIMENTS

Herein, we extend our experiments for the pedagogical case study. In what follows, we provide numerical illustrations for a series of binary optimization problems. For the purpose of decoding candidate solutions from the continuous unified space ( $\mathbf{X} = [0, 1]^{D_{unified}}$ ) to the respective discrete/binary representations, a variable (of an encoded solution) is assigned a value of 0 if its value in the unified space is less than or equal to 0.5. Else, it is assigned a value of 1. Unless otherwise stated, the parameter settings for SBX crossover, polynomial mutation, population size, etc., remain the same as in Section VI-A of the paper.

#### A. Case 1: Trap-5 + hamming distance minimization (HDM) problems

In this particular study, we conduct experiments to solve the trap-5 problem alongside a series of hamming distance minimization (HDM) binary problems using MFEA-II. Herein, we generate a range of target strings for the HDM problems by sampling them from a Bernoulli parameter  $\rho$ , whereby a single bit takes the value 1 with probability  $\rho$  (and the value 0 with probability  $1 - \rho$ ). For each successive experiment, the Bernoulli parameter is gradually decremented from  $\rho = 1.00$  to  $\rho = 0.00$ . It is noteworthy that if  $\rho$  is higher, the HDM problems behave similar to the onemax problem and thus bear higher degrees of intersection (overlap), in terms of the global optimum, with the trap-5 problem. In contrast, if  $\rho$  is lower, the HDM problems behave more like the zeromax problem. Hence, the global optimum of these HDM problems are further away from the global optimum of trap-5. For this set of experiments, we solve 100 dimensional variants of the trap-5 and the HDM problems.

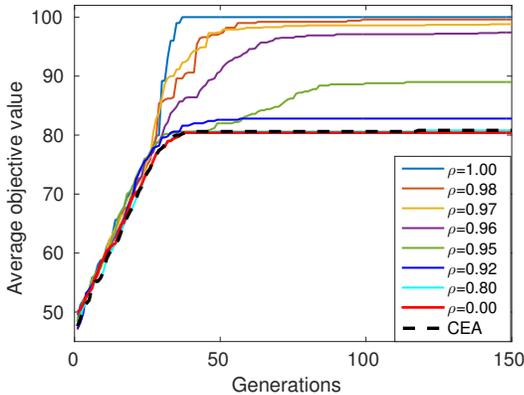


Fig. 3. Trap-5 convergence trends.

The convergence trends for trap-5 are shown in Figure 3 (results are averaged over 30 independent runs). From Figure 3, it can be seen that the performance of MFEA-II is superior to CEA when multitasked with HDM problems sampled from higher  $\rho$ 's (e.g.  $\rho = 1.00$  to  $\rho = 0.92$ ). Under the positive influence of genetic transfers from these HDM tasks, the MFEA-II is unlikely to get trapped in the deceptive local optimum of trap-5. However, as  $\rho$  decreases, the convergence trends of

MFEA-II gradually drops and approaches that of the CEA (i.e., the MFEA-II performs identical to the single tasking CEA). Most importantly, no negative transfer is observed even in the extreme case of multitasking trap-5 alongside a HDM problem with  $\rho = 0.00$ . Further, the learned  $rmp$  trends in Figures 4 and 5 reveal that MFEA-II is adept at capturing the inter-task relationships online and adapts genetic transfers in a manner that leverages on positive transfers between overlapping tasks (dictated by higher  $\rho$ 's) while suppressing any deleterious effects of negative exchanges otherwise. Notice that for distant (unrelated) tasks (e.g.  $\rho = 0.80$ ), the  $rmp$  values learned (throughout the multitasking search) are near zero, implying that the amount of resources wasted in transferring and evaluating solutions across unrelated tasks is negligible.

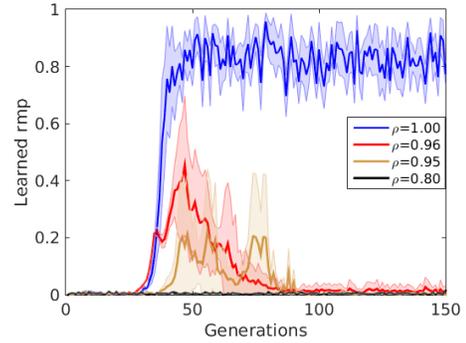


Fig. 4. Learned  $rmp$  trends across generations. The shaded regions span one-half standard deviation on either side of the mean.

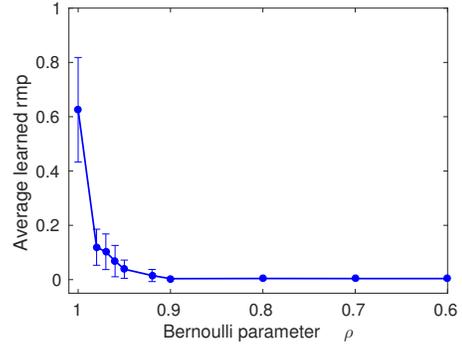


Fig. 5. Average learned  $rmp$  trend between trap-5 and a series of HDM problems dictated by  $\rho$ . The error bars span one-half standard deviation on either side of the mean.

#### B. Case 2: Inverted trap-5 + HDM problems

In this case study, we invert the roles of zeros and ones in the definition of the original trap-5 function such that the global optimum is changed to a string of all zero bits. The experimental setup is similar to Section III-A above.

In contrast to trap-5, the HDM problems sampled from lower  $\rho$ 's (e.g.  $\rho = 0.00$  to  $\rho = 0.08$ ) would, herein, induce (higher) positive influence towards improving the search of the inverted trap-5 problem - showing greater potential for fruitful genetic transfers as demonstrated by Figure 6. The extent of

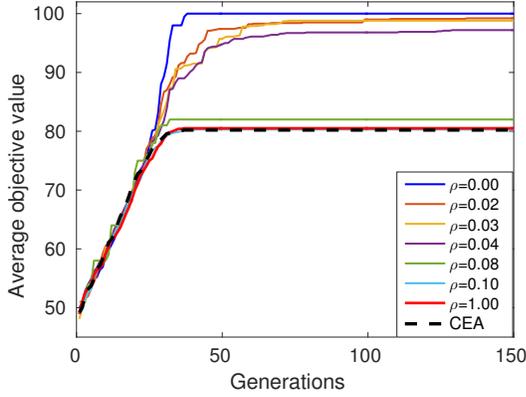


Fig. 6. Inverse trap-5 convergence trends.

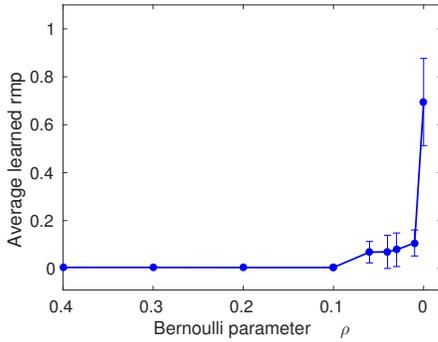


Fig. 7. Average learned  $rmp$  trend between the inverse trap-5 and a series of HDM problems dictated by  $\rho$ . The error bars span one-half standard deviation on either side of the mean.

genetic transfers between the inverted trap-5 problem and a series of HDM problems (with varying  $\rho$ 's) is mandated by the learned  $rmp$  trend in Figure 7.

### C. Case 3: $K$ randomly generated HDM tasks

In this study, we consider solving  $K$  tasks, each minimizing the hamming distance to a given target string generated at random ( $\rho = 0.5$ ). Herein, we set  $K = 10$  and the dimensionality of each HDM task is set to  $D = 50$ . Figure 8 shows

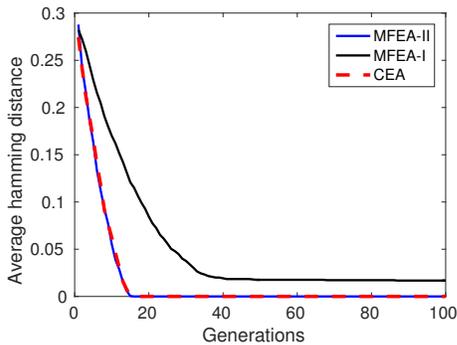


Fig. 8. Averaged performance of MFEA-II, MFEA-I and CEA on 10 randomly generated HDM tasks.

the convergence trends averaged across all the 10 tasks. It can

be observed the MFEA-II and the single tasking CEA perform identical to each other. This is primarily because the randomly generated tasks tend to bear low degrees of intersection of global optima. As such, their population datasets may not overlap as the search progresses and genetic transfers could in fact be detrimental. While MFEA-II prevents the harmful exchange of genetic material, we observe that MFEA-I is plagued by significant negative transfers. The convergence trend from Figure 8 shows that the lack of an online transfer parameter learning strategy in MFEA-I hampers its performance on the 10 HDM tasks. The ability of MFEA-II to overcome this issue marks the key contribution of the present paper.

Figure 9 shows a few representative  $rmp$  trends learned between pairs of randomly generated tasks. As is revealed by the relatively low (near zero)  $rmp$  trends, the learning module of MFEA-II is able to figure out that there is no utility for genetic transfers herein, thereby ensuring that the convergence characteristics are not hampered due to harmful transfers.

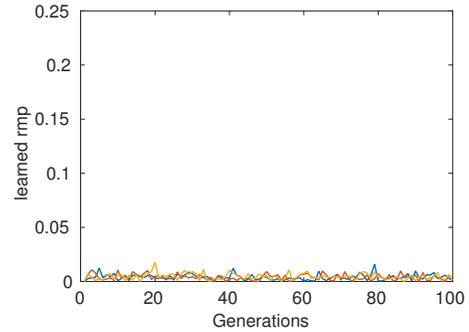


Fig. 9. Few representative  $rmp$  curves between pairs of randomly generated HDM tasks. Other inter-task  $rmp$  values showcase similar trends.

### D. Case 4: LeadingOnes + TrailingOnes problem

For this particular study, we consider solving the LeadingOnes and TrailingOnes problems simultaneously using MFEA-II. The LeadingOnes problem essentially aims to maximize the number of consecutive 1-bits counting from the left of a solution string. In contrast, the TrailingOnes problem starts the search by scanning the solution from the right towards the left. Both the problems share the same global optimum (i.e., the string with all 1's). For this experimental study, we set the problem dimensionality and the population size to  $D = 50$  and  $N = 150$  for each of the tasks, respectively.

Notably, while the problems do share a common global optimum, transfers in the early stage of the optimization phase is unlikely to be beneficial since each of the tasks begin the search from opposite ends of the search space. In particular, Figure 10 demonstrates that MFEA-II is able to identify this and prescribes lower (near zero)  $rmp$  values during the initial stages. As the search progresses and the tasks becomes more similar, MFEA-II gradually learns higher  $rmp$  values in order to facilitate fruitful knowledge transfers. As shown by Figure 11, the transfer adaptation strategy of MFEA-II enables both the tasks to progress towards the common global optimum. In contrast, the convergence trends of MFEA-I are

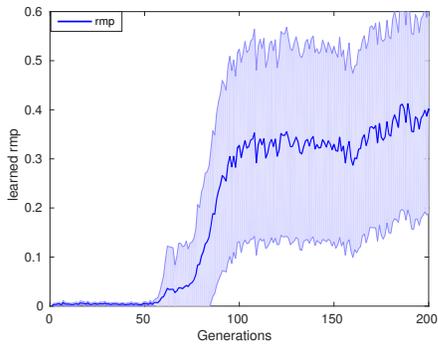


Fig. 10. Learned *rmp* trend between LeadingOnes and TrailingOnes problems over successive generations. The shaded regions span one-half standard deviation on either side of the mean.

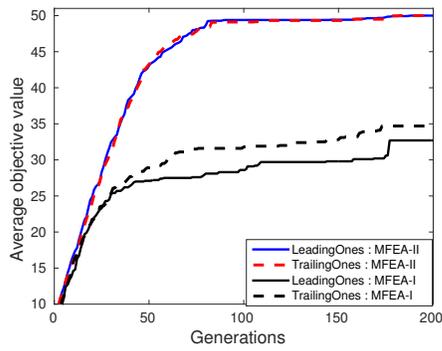


Fig. 11. Convergence trends of MFEA-II and MFEA-I.

significantly hampered due to continuous (unregulated) inter-task genetic transfers.