Multifactorial Evolutionary Algorithm With Online Transfer Parameter Estimation: MFEA-II

Kavitesh Kumar Bali, Yew-Soon Ong, *Fellow, IEEE*, Abhishek Gupta, and Puay Siew Tan

***Abstract*—Humans rarely tackle every problem from scratch. Given this observation, the motivation for this paper 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 intertask synergies (as is often the case with general black-box optimization), the threat of predominantly *negative transfer* prevails. Susceptibility to negative intertask 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 the 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, general optimization intelligence (GOI), intertask synergies, multifactorial optimization, 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 TO 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; reoptimization of the source task(s) is irrelevant. Solving multiple tasks concurrently is thus, beyond the scope of sequential TO 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,

Manuscript received December 18, 2017; revised June 6, 2018, October 31, 2018, and March 4, 2019; accepted March 13, 2019. Date of publication March 22, 2019; date of current version January 29, 2020. This work was supported in part by the Singapore Institute of Manufacturing Technology-Nanyang Technological University Joint Laboratory and Collaborative Research Programme on Complex Systems and in part by the National Research Foundation Singapore through AI Singapore Programme under Award AISG-RP-2018-00. *(Corresponding author: Abhishek Gupta.)*

K. K. Bali is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: bali0001@e.ntu.edu.sg).

Y.-S. Ong is with the Data Science and Artificial Intelligence Research Centre, School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: asysong@ntu.edu.sg).

A. Gupta and P. S. Tan are with the Singapore Institute of Manufacturing Technology, Agency for Science, Technology and Research, Singapore 138634 (e-mail: abhishek_gupta@simtech.a-star.edu.sg; pstan@simtech.a-star.edu.sg).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2019.2906927

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 intertask 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 intertask 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 intertask interactions. It is noteworthy that the ideas described herein directly encompass a previously proposed multifactorial EA (MFEA) for multitasking [18]. In comparison to the MFEA, our main contributions in this paper are twofold.

1) 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 intertask relationships.
2) 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 subtrees 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 EA 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. The 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 this 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 intertask 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 maybe difficult to deal with [31].

In contrast, 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 (FNNs) 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 intertask 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 intertask 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 $\{T_1, T_2, \ldots, T_K\}$ to be solved concurrently. The search space dimensionality of each of the optimization tasks is given by $D_1, D_2, \ldots, D_K$, respectively. In such a scenario, we can define a *unified space* $\boldsymbol{X}$ of dimensionality $D_{\text{unified}} = \max\{D_1, D_2, \ldots, D_K\}$, 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 $D_{\text{unified}} = (D_1 + D_2 + \cdots + D_K)$, is twofold.

1) It helps circumvent the challenges associated with the curse of dimensionality when solving several tasks with multidimensional search spaces simultaneously.
2) 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 $\boldsymbol{X}$ to be limited to the range $[0, 1]^{D_{\text{unified}}}$, which serves as a *continuized* unified space into which all candidate solutions are mapped (*encoded*). Accordingly, while tackling the kth task, a subset of D_k variables are extracted from a candidate solution $x \in \boldsymbol{X}$ 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 $k \in \{1, 2, \ldots, K\}$, let the (sub)populations associated with the kth 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 kth task. Note that we assumed x to be a candidate solution in the *unified search space* $\boldsymbol{X}$. 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 $H = \{x | x \in \boldsymbol{X}, f_k(x) > f_k'\}$ 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 $p^1(x, t), p^2(x, t), \ldots, p^K(x, t)$, respectively. Accordingly, in the process of facilitating *intertask* interactions, the offspring population generated for the kth task at the t-th iteration is considered to be drawn from the following mixture distribution:

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

The finite mixture $q^k(x, t)$ is a linear combination of all K available distributions, with mixture coefficients given by the α's [43]. Note that $\alpha_k + \sum_{j \neq k} \alpha_j = 1$ and α's ≥ 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 (1) [8], [44]. The degree of mixing (i.e., the amount of knowledge exchange) is mandated by the mixture coefficients α's. It is noteworthy that in the absence of prior knowledge about the intertask 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 (α'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 $N \to \infty$. 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 $\boldsymbol{X}$ for all $k \in \{1, 2, \ldots, K\}$. 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:

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

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 (μ, λ) selection strategy whereby μ denotes the number of parents, and λ represents the number of offspring. Under this selection mechanism, the top μ individuals out of λ 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 kth 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 (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 (1), the (μ, λ) parent selection scheme can be 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 (1), (3), and (4).

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

Proof: Our deductions follow a similar style as [47]. At any iteration $t > 0$, the μ 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 (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 (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 (1) for the left-hand side of (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, (5) implies $\int_{f_k(x)\geq\beta^k(t)} p^k(x, t) \cdot dx = 1$, which on combining with (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 (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 β^k can not be greater than f_k^* implies that there exists a limit $\lim_{t\to\infty} \beta^k(t) = f_k'$ 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 (1) and (3), note

$$p^k(x, t) \geq p^k(x, 0)\left[\frac{\alpha_k}{\theta}\right]^t \ \forall\, x : f_k(x) > f_k'. \quad (11)$$

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

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

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

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

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

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

■

IV. 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 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 Σ. Additionally, let the mean and covariance of the offspring population be m_c and Σ_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 δ_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 σ_i^2 is the induced variance of the ith variable during offspring creation. The specified high probability of creating offspring near the parents implies that $\sigma_i^2 << \Sigma_{(i,i)}$. 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 $\{T_1, T_2, \ldots, T_K\}$ simultaneously, where each task is seen as an added *factor* influencing the evolution of the population. The subpopulation associated with the kth task is denoted as P^k. Given this background, certain terminologies commonly encountered in the context of the MFEA are as follows.

1) *Definition 1 (Skill Factor):* The *skill factor* τ_i of the ith individual is the one task, amongst all K tasks in the multitasking environment, with which the individual is associated.

2) *Definition 2 (Scalar Fitness):* The *scalar fitness* of the ith individual in a multitasking environment is given by $\varphi_i = 1/r^i_{\tau_i}$, where $r^i_{\tau_i}$ is the rank of the ith individual on task τ_i.

Algorithm 1 Pseudocode of a Basic MFEA

```
1  Randomly sample $N \cdot K$ individuals in $\boldsymbol{X}$ to form initial population $P(0)$;
2  for every individual $p_i$ in $P(0)$ do
3      Assign skill factor $\tau_i = \mod (i, K) + 1$, 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 rand$\leq$ 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     $t = t + 1$;
```

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 $\boldsymbol{X}$. 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 *intratask crossover* between parents of the same skill factor, *intertask 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.

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 rmp $\in [0, 1]$. During intertask 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 kth 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 intertask 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 1: Under the stated assumptions of parent-centric evolutionary operators, the offspring mixture distribution $p_c^k(x, t)$ of the kth task in the MFEA can be expressed as

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

Proof: See the supplementary material. ■

Note the resemblance between (15) and (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 2: Under the large population assumption (i.e., $N \rightarrow \infty$), the MFEA with parent-centric evolutionary operators and (μ, λ) 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 kth task in the multitasking environment, it suffices to ensure that $\alpha_k > \theta$. With reference to Lemma 1, the mixture coefficient α_k corresponds to

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

Since $0 \leq \text{rmp} \leq 1$, it follows that $0.5 < \alpha_k \leq 1$, and thus α_k is indeed greater than θ. The above holds for all k. ■

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 intertask relationships, tuning an appropriate rmp (via trial and error) thus becomes burdensome. Essentially, poor prescriptions of rmp risk the possibility of either predominantly negative intertask interactions, or the potential loss of fruitful knowledge exchange.

In this section, we study the possible deleterious effects of negative intertask 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 kth 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:

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

Qualitatively speaking, the optimum solution(s) of the kth 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 tth 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 intertask 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 (μ, λ) selection with $\theta \leq 0.50$, $\beta^{k,ST}(t+1)$ satisfies

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

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

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

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

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

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

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

Equation (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 (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 1, this can easily be accomplished by assigning rmp $= 0$. However, such a naive strategy prohibits intertask 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 intertask 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 (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

$$\mathrm{RMP} = \begin{bmatrix} \mathrm{rmp}_{1,1} & \mathrm{rmp}_{1,2} & \cdot & \cdot \\ \mathrm{rmp}_{2,1} & \mathrm{rmp}_{2,2} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix} \tag{22}$$

where $\mathrm{rmp}_{j,k} = \mathrm{rmp}_{k,j}$ captures the synergy (or the extent of transfer) between the jth and kth tasks. Accordingly, it is naturally enforced that $\mathrm{rmp}_{j,j} = 1, \forall j$. 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 nonuniform intertask 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 kth 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, (15) can be rewritten as

$$g_c^k(x,t) = \left[1 - \frac{0.5}{K} \cdot \sum_{k \neq j} \mathrm{rmp}_{k,j}\right] \cdot g^k(x,t) + \frac{0.5}{K} \sum_{j \neq k} \mathrm{rmp}_{k,j} \times g^j(x,t) \tag{23}$$

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 $k \in \{1, 2, \ldots, K\}$.

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 $k \in \{1, 2, \ldots, K\}$. From here onward, consider the parameter θ $(= \mu/\lambda)$ 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:

$$\max{}_{\mathrm{RMP}} \sum_{k=1}^{K} \sum_{i=1}^{N/2} \log\, g_c^k(x_{ik}, t) \tag{24}$$

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

We claim that the data-driven learning paradigm of (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

$$\mathrm{KL}(p||g) = \int_{\boldsymbol{X}} p(x) \cdot \big[\log\, p(x) - \log\, g(x)\big] \cdot dx. \tag{25}$$

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

Theorem 3: Maximizing the likelihood function in (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 $\{T_1, T_2, \ldots, T_k, \ldots, T_K\}$.

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

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

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

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

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

$$\max{}_{\mathrm{RMP}} \sum_{k=1}^{K} \int_{\boldsymbol{X}} p^k(x,t) \cdot \log\, g_c^k(x,t) \cdot dx. \tag{28}$$

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

$$\min{}_{\mathrm{RMP}} \sum_{k=1}^{K} \int_{\boldsymbol{X}} p^k(x,t) \cdot \Big[\log\, p^k(x,t) - \log\, g_c^k(x,t)\Big] \cdot dx. \tag{29}$$

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

$$\min{}_{\mathrm{RMP}} \sum_{k=1}^{K} KL\Big(p^k(x,t)||g_c^k(x,t)\Big). \tag{30}$$

In other words, the mathematical program of (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. ■

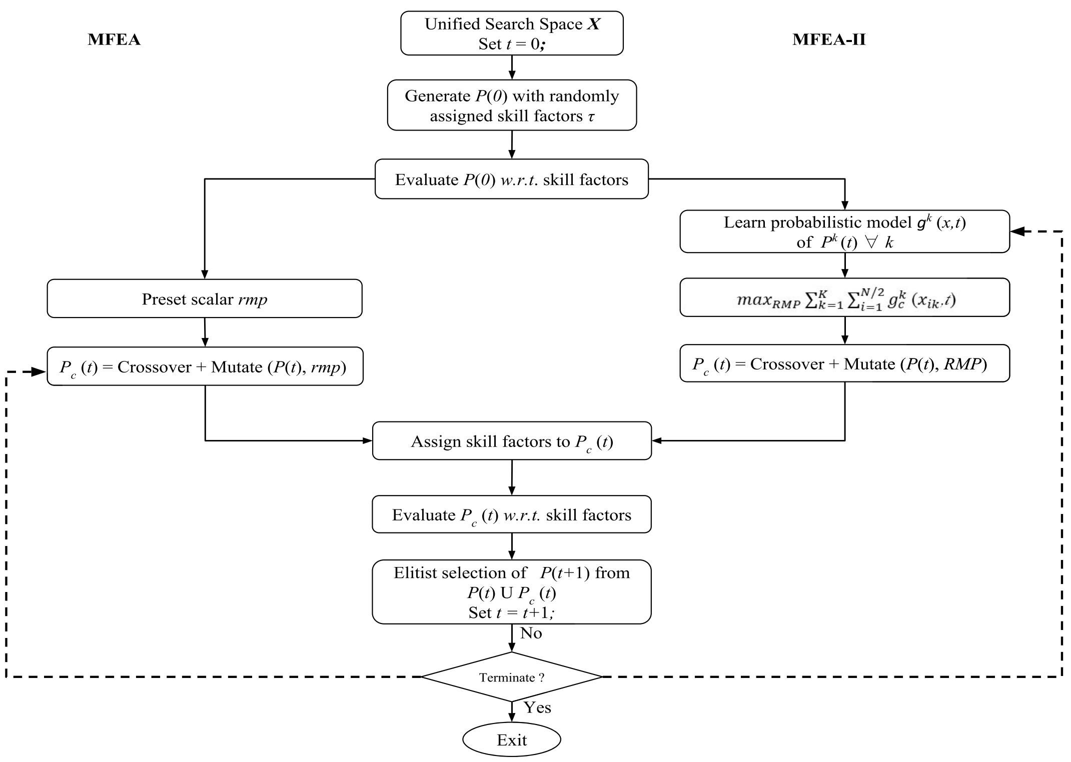
MFEA
MFEA-II
Unified Search Space X
Set t = 0;
Generate P(0) with randomly assigned skill factors τ
Evaluate P(0) w.r.t. skill factors
Learn probabilistic model g^k (x,t) of P^k(t) ∀ k
Preset scalar rmp
max_RMP Σ_{k=1}^K Σ_{i=1}^{N/2} g_c^k (x_ik, t)
P_c (t) = Crossover + Mutate (P(t), rmp)
P_c (t) = Crossover + Mutate (P(t), RMP)
Assign skill factors to P_c (t)
Evaluate P_c (t) w.r.t. skill factors
Elitist selection of P(t+1) from P(t) U P_c (t)
Set t = t+1;
No
Terminate ?
Yes
Exit

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.

Theorem 3 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 $P^k(t) \in \{P^1(t), P^2(t), \ldots, P^K(t)\}$. 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 cancelation of intertask 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.

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 Fig. 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 (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 $P^k(t)$, ∀
  $k \in \{1, 2, ..., K\}$.
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 intertask 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 intertask 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 (CEAs) executing single-tasking in parallel. The ability of the MFEA-II to identify

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 $rand \leq rmp_{\tau_i,\tau_j}$ 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 $\tau_j$;
```

discrepancies and interpret similarities across multiple tasks on the fly is exemplified through subsequent experimental studies.

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. First, we consider the basic counterparts, i.e., the present day multifactorial evolutionary algorithm (denoted as MFEA-I) [18] and the standalone 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[1] 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 as follows.

1) Continuized unified representation with range $[0, 1]^{D_{\text{unified}}}$.
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) PM 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) *rmp:* 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 PM 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 ($\boldsymbol{X} = [0, 1]^{D_{\text{unified}}}$) to the discrete/binary space, a variable is assigned a value of 0 if its encoded counterpart is less than or equal to ≤ 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 five nonoverlapping bits. Further, a 5-bit trap function is applied to each of the groups, and the contribution of each group toward the combined objective function is given as follows:

$$\text{trap}_5(u) = \begin{cases} 5 & \text{if } u = 5 \\ (4 - u) & \text{if } u < 5 \end{cases} \tag{31}$$

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 $(2^{(d/5)} - 1)$ 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

[1]https://openai.com/

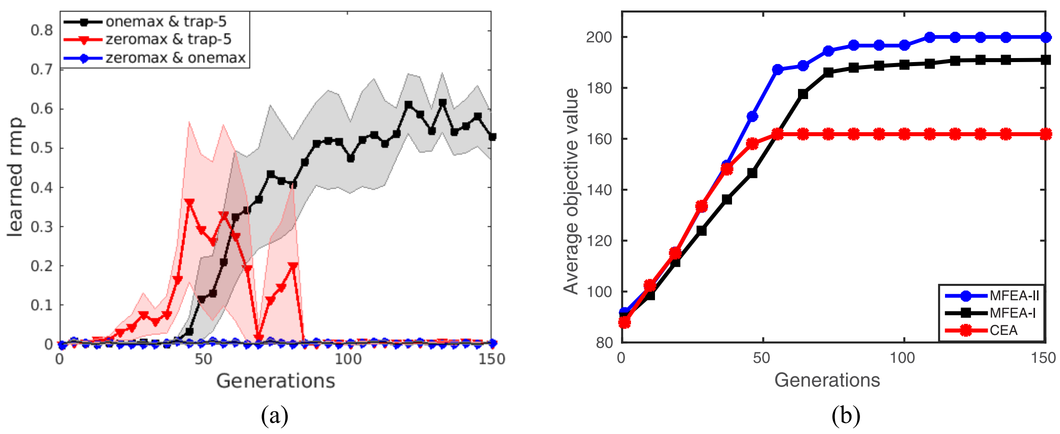

Fig. 2. Left: Pairwise rmp's learned between distinct tasks over successive generations. The shaded regions span one half standard deviation on either side of the mean. Right: Convergence trends (averaged over 30 independent runs) achieved by all the algorithms on the trap-5 problem. As shown, both multitasking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

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 intertask 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 rmp), would be susceptible to negative transfers.

In this paper, 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 Fig. 2. Note that a 3×3 symmetrical RMP matrix is learned for MFEA-II, capturing the pairwise intertask similarities over successive generations. The learned RMP matrix elements are presented in Fig. 2(a).

Fig. 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 Fig. 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 Fig. 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 intertask 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 toward 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 Fig. 2(b)].

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

C. Synthetic Continuous Benchmark Functions

In the next set of examples, we showcase the rmp learned given *known* intertask relationships between task pairs. To this end, we utilize some of the CEC composite minimization benchmark functions comprising of high, medium, and low intertask similarities [25]. The intertask similarity measure between a pair of tasks (R_s) is based on the Spearmans 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 intertask 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.

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	

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 T_1	MFEA-II T_2	MFEA-I T_1	MFEA-I T_2	CEA T_1	CEA T_2
1 CI+HS	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	82.35
	0.00	0.00	0.00	0.00	0.00	11.51
2 CI+MS	**0.00**	**0.00***	**0.00**	5.57	18.82	82.17
	0.00	0.00	0.00	7.26	0.92	12.17
3 CI+LS	**19.98**	**1.67E+03***	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	**55.42**	**4.08***	58.51	13.10	65.99	83.89
	29.91	9.73	47.99	34.33	32.52	13.06
5 NI+MS	**0.00**	**20.61**	**0.00**	21.08	**0.00**	31.52
	0.00	1.19	0.00	1.72	0.00	4.96

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.

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 intertask 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 intertask 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 intertask 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. Fig. 3 reveals that the MFEA-II successfully deciphers the underlying intertask similarities *online*. Notice from Fig. 3(a) and (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 intertask similarities ($R_s \approx 1$). Similarly, relatively lower rmp values are learned for the problem pairs NI+MS and CI+LS with lower intertask similarity [see Fig. 3(c) and (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.

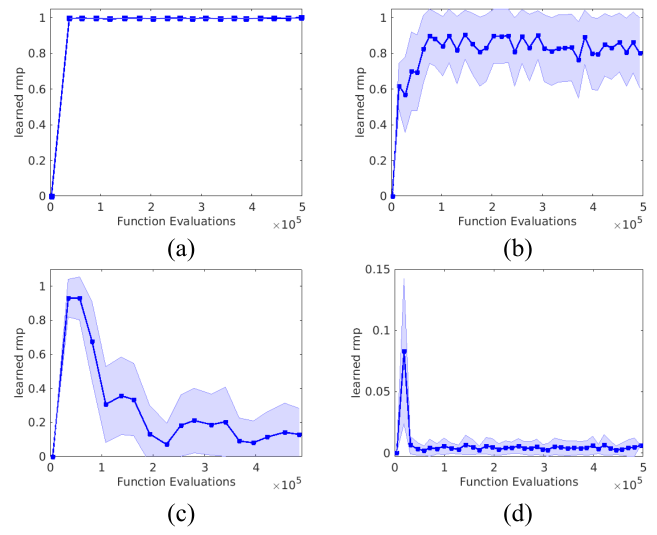

Fig. 3. Learned pairwise rmp values between pairs of tasks achieved by MFEA-II on different problem sets characterized by different degrees of intertask similarities. The plots are averaged over 30 independent runs with the shaded regions spanning one-half standard deviation on either side of the mean. (a) CI+HS. (b) NI+HS. (c) NI+MS. (d) CI+LS.

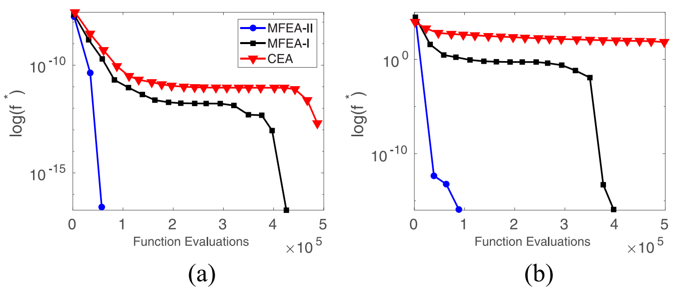

Fig. 4. Convergence trends (averaged over 30 independent runs) achieved by the three tested algorithms on the problem set CI+HS with high degrees of intertask similarity. (a) CI+HS–T_1. (b) CI+HS–T_2.

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 Fig. 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 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 Fig. 3(a).

D. 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 toward 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 1.0 m 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 1.0 m.

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 paper 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—θ_1, angular velocity of longer pole—$\dot{\theta}_1$, angle of shorter pole—θ_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] and [70].

The dynamical system is evaluated using the fourth-order Runge–Kutta numerical integration scheme with step size $\tau = 0.01$ s. 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 FNN controller, with ten 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 s. 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 4.8 m 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 0.02 s. This is equivalent to over 30 min of simulated time.

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%

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.

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 ($l_s = 0.70$ m), 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 0.45 to 0.70 m. 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.

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

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

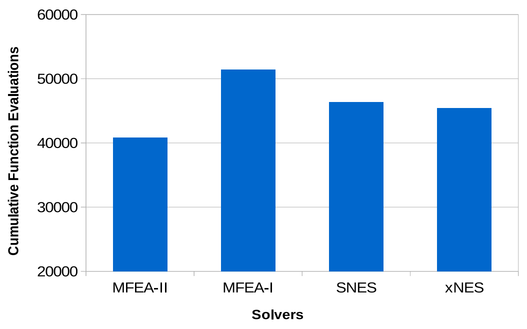

Fig. 5. Comparison of total number of function evaluations consumed by each of the algorithms in solving all the six different tasks.

Apart from MFEA-II, the success rate of all other methods is significantly low on the increasingly challenging variants T_5 ($l_s = 0.65$ m) and T_6 ($l_s = 0.70$ m). 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 $l_s = 0.70$ m.

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 Fig. 5.

Further, Fig. 6 shows the pairwise $\text{rmp}_{j,k}$ 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 intertask 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, Fig. 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. 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 $T_4/T_5/T_6$. Fig. 6 shows that MFEA-II precisely captures this trend. In other words, as the expected intertask similarity decreases, so do the learned rmp values.

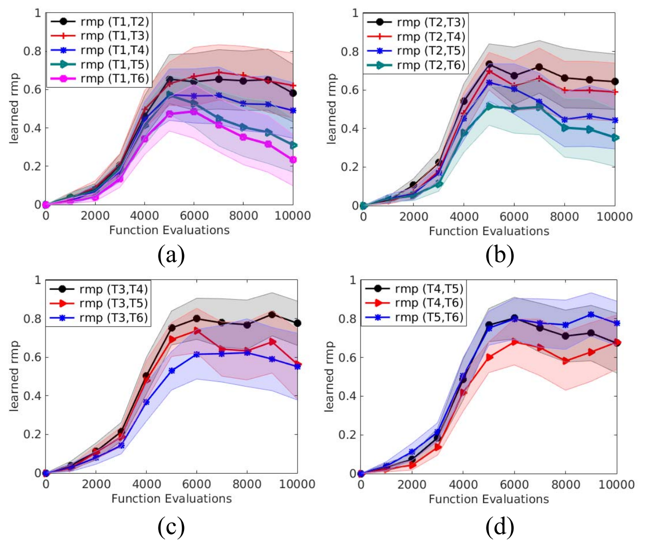

Fig. 6. Pairwise rmp's of MFEA-II, learned between constitutive tasks of different short pole lengths. The plots are averaged over 30 independent runs and the shaded regions span one-half standard deviation on either side of the mean. (a) Set_1. (b) Set_2. (c) Set_3. (d) Set_4.

VII. CONCLUSION

In this paper, we proposed a novel online transfer parameter estimation scheme for evolutionary multitasking, capable of learning and exploitation of intertask 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 intertask similarities *online* in a purely data-driven manner. The algorithmic contributions of MFEA-II can be summarized from two perspectives.

1) The transfer parameter takes the form of a symmetric matrix for effectively multitasking across more than two tasks with possibly diverse intertask relationships.
2) 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 intertask 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 intertask 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.

References

[1] S. Thrun, "Is learning the *n*-th thing any easier than learning the first?" in *Proc. Adv. Neural Inf. Process. Syst.*, 1996, pp. 640–646.

[2] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[3] M. Iqbal, W. N. Browne, and M. Zhang, "Extracting and using building blocks of knowledge in learning classifier systems," in *Proc. 14th Annu. Conf. Genet. Evol. Comput.*, 2012, pp. 863–870.

[4] M. Iqbal, W. N. Browne, and M. Zhang, "Reusing building blocks of extracted knowledge to solve complex, large-scale Boolean problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 465–480, Aug. 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 Trans. Evol. Comput.*, vol. 21, no. 4, pp. 569–587, Aug. 2017.

[6] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, 1997.

[7] Y.-S. Ong and A. Gupta, "Evolutionary multitasking: A computer science view of cognitive multitasking," *Cogn. Comput.*, 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 Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 51–64, Feb. 2018.

[9] A. Gupta and Y.-S. Ong, *Memetic Computation: The Mainspring of Knowledge Transfer in a Data-Driven Optimization Era*, vol. 21. Cham, Switzerland: Springer, 2018.

[10] S. Cabi *et al.*, "The intentional unintentional agent: Learning to solve many continuous control tasks simultaneously," in *CoRL PMLR*, 2017, pp. 207–216.

[11] P. Cunningham and B. Smyth, "Case-based reasoning in scheduling: Reusing solution components," *Int. J. Prod. Res.*, vol. 35, no. 11, pp. 2947–2962, 1997.

[12] M. Kaedi and N. Ghasem-Aghaee, "Biasing Bayesian optimization algorithm using case based reasoning," *Knowl. Based Syst.*, 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 *Proc. Int. Conf. Parallel Problem Solving Nat.*, 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 Comput.*, 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," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 501–514, Aug. 2018.

[16] B. Da, A. Gupta, and Y.-S. Ong, "Curbing negative influences online for seamless transfer evolutionary optimization," *IEEE Trans. Cybern.*, to be published.

[17] K. Swersky, J. Snoek, and R. P. Adams, "Multi-task Bayesian optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2004–2012.

[18] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 343–357, Jun. 2016.

[19] A. Gupta, Y.-S. Ong, L. Feng, and K. C. Tan, "Multiobjective multifactorial optimization in evolutionary multitasking," *IEEE Trans. Cybern.*, vol. 47, no. 7, pp. 1652–1665, Jul. 2017.

[20] L. Feng *et al.*, "Evolutionary multitasking via explicit autoencoding," *IEEE Trans. Cybern.*, to be published.

[21] A. Gupta and Y.-S. Ong, "Back to the roots: Multi-*X* evolutionary computation," *Cogn. Comput.*, vol. 11, no. 1, pp. 1–17, 2019.

[22] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multitasking for evolutionary optimization of expensive problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 44–58, Feb. 2019.

[23] J. Tang, Y. Chen, Z. Deng, Y. Xiang, and C. P. Joy, "A group-based approach to improve multifactorial evolutionary algorithm," in *Proc. IJCAI*, 2018, pp. 3870–3876.

[24] R.-T. Liaw and C.-K. Ting, "Evolutionary manytasking optimization based on symbiosis in biocoenosis," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 1–9.

[25] B. Da *et al.*, "Evolutionary multitasking for single-objective continuous optimization: Benchmark problems, performance metric, and baseline results," Rep., 2016.

[26] E. V. Bonilla, K. M. Chai, and C. K. I. Williams, "Multi-task Gaussian process prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 153–160.

[27] M. Pelikan, M. W. Hauschild, and F. G. Lobo, "Estimation of distribution algorithms," in *Springer Handbook of Computational Intelligence*. Berlin, Germany: 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1717–1724.

[29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. Adv. Neural Inf. Process. Syst.*, 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," *Proc. IEEE*, vol. 104, no. 1, pp. 148–175, Jan. 2016.

[31] M. Zaefferer and T. Bartz-Beielstein, "Efficient global optimization with indefinite kernels," in *Proc. Int. Conf. Parallel Probl. Solving Nat.*, 2016, pp. 69–79.

[32] H. Mühlenbein, M. Gorges-Schleuter, and O. Krämer, "Evolution algorithms in combinatorial optimization," *Parallel Comput.*, 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 *Proc. Int. Conf. Parallel Probl. Solving Nat.*, 1994, pp. 249–257.

[34] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 378–393, Jun. 2014.

[35] Q. Chen, X. Ma, Y. Sun, and Z. Zhu, "Adaptive memetic algorithm based evolutionary multi-tasking single-objective optimization," in *Proc. Asia–Pac. Conf. Simulat. Evol. Learn.*, 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 *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2017, pp. 2266–2273.

[37] Y.-W. Wen and C.-K. Ting, "Parting ways and reallocating resources in evolutionary multitasking," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2017, pp. 2404–2411.

[38] Y.-W. Wen and C.-K. Ting, "Learning ensemble of decision trees through multifactorial genetic programming," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 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 Process. Lett.*, vol. 47, no. 3, pp. 993–1009, 2017.

[40] A. Gupta, Y. S. Ong, B. Da, L. Feng, and S. D. Handoko, "Measuring complementarity between function landscapes in evolutionary multitasking," in *Proc. IEEE Congr. Evol. Comput.*, 2016, pp. 1–8.

[41] K. K. Bali, A. Gupta, L. Feng, Y. S. Ong, and T. P. Siew, "Linearized domain adaptation in evolutionary multitasking," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2017, pp. 1295–1302.

[42] A. Gupta, J. Mańdziuk, and Y.-S. Ong, "Evolutionary multitasking in bi-level optimization," *Complex Intell. Syst.*, vol. 1, nos. 1–4, pp. 83–95, 2015.

[43] P. Smyth and D. Wolpert, "Stacked density estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, 1998, pp. 668–674.

[44] P. Larrañaga and J. A. Lozano, Eds., *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, vol. 2. New York, NY, USA: Springer, 2001.
[45] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, vol. 31. New York, NY, USA: Springer, 2013.
[46] H.-G. Beyer, *The Theory of Evolution Strategies*. New York, NY, USA: Springer, 2013.
[47] Q. Zhang and H. Muhlenbein, "On the convergence of a class of estimation of distribution algorithms," *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 127–136, Apr. 2004.
[48] Y. Chow and H. Yeicher, *Probability Theory*, 3rd ed. Philadelphia, PA, USA: Soc. Ind. Appl. Math., 1997.
[49] K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization," *Evol. Comput.*, vol. 10, no. 4, pp. 371–395, Dec. 2002.
[50] R. B. Agrawal, K. Deb, and R. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, no. 2, pp. 115–148, 1995.
[51] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Comput. Sci. Informat.*, vol. 26, no. 4, pp. 30–45, 1996.
[52] R. Hinterding, "Gaussian mutation and self-adaption for numeric genetic algorithms," in *Proc. IEEE Int. Conf. Evol. Comput.*, vol. 1, 1995, p. 384.
[53] K. Deb, D. Joshi, and A. Anand, "Real-coded evolutionary algorithms with parent-centric recombination," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 1, 2002, pp. 61–66.
[54] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Stat.*, vol. 22, no. 1, pp. 79–86, 1951.
[55] D. Pardoe and P. Stone, "Boosting for regression transfer," in *Proc. 27th Int. Conf. Int. Conf. Mach. Learn.*, 2010, pp. 863–870.
[56] F. Gomez, J. Schmidhuber, and R. Miikkulainen, "Accelerated neural evolution through cooperatively coevolved synapses," *J. Mach. Learn. Res.*, vol. 9, pp. 937–965, May 2008.
[57] D. Wierstra *et al.*, "Natural evolution strategies," *J. Mach. Learn. Res.*, 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 *Proc. 13th Annu. Conf. Genet. Evol. Comput.*, 2011, pp. 845–852.
[59] T. Glasmachers, T. Schaul, S. Yi, D. Wierstra, and J. Schmidhuber, "Exponential natural evolution strategies," in *Proc. 12th Annu. Conf. Genet. Evol. Comput.*, 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. (Nov. 2017). *Natural Evolution Strategies*. [Online]. Available: http://people.idsia.ch/tom/nes.html
[62] M. López-Ibánez, J. Dubois-Lacoste, T. Stützle, and M. Birattari, "The IRACE package, iterated race for automatic algorithm configuration," IRIDIA, Université Libre de Bruxelles, Brussels, Belgium, Rep. TR/IRIDIA/2011-004, 2011.
[63] K. Sastry, D. E. Goldberg, and M. Pelikan, "Limits of scalability of multiobjective estimation of distribution algorithms," in *Proc. IEEE Congr. Evol. Comput.*, vol. 3, 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 *Proc. IEEE Region 10 Conf. (TENCON)*, 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 *Proc. Int. Conf. Neural Inf. Process.*, 2016, pp. 37–46.
[66] R. S. Sutton, "Temporal credit assignment in reinforcement learning," Ph.D. dissertation, Dept. Comput. Inf. Sci., Univ. Massachusetts, Boston, MA, USA, 1984.
[67] F. J. Gomez and R. Miikkulainen, "Solving non-Markovian control tasks with neuroevolution," in *Proc. IJCAI*, vol. 99, 1999, pp. 1356–1361.
[68] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evol. Comput.*, vol. 10, no. 2, pp. 99–127, 2002.
[69] C. Igel, "Neuroevolution for reinforcement learning using evolution strategies," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 4, 2003, pp. 2588–2595.
[70] A. P. Wieland, "Evolving neural network controllers for unstable systems," in *Proc. Seattle Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 2, 1991, pp. 667–673.

Kavitesh Kumar Bali received the M.Sc. degree in computer science from the University of the South Pacific, Suva, Fiji, in 2016. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, Nanyang Technological University, Singapore.

His current research interests include evolutionary computation, transfer optimization, and machine learning.

Yew-Soon Ong (M'99–SM'12–F'18) received the Ph.D. degree in artificial intelligence in complex design from the University of Southampton, Southampton, U.K., in 2003.

He is currently a Professor with the School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore, where he is also the Director of the Data Science and Artificial Intelligence Research Center and Singtel-NTU Cognitive and Artificial Intelligence Corporate Laboratory. His current research interest include artificial intelligence spans across memetic computing, optimization intelligence, and machine learning.

Dr. Ong was a recipient of the several IEEE outstanding paper awards and listed as a Thomson Reuters Highly Cited Researcher and among the World's Most Influential Scientific Minds. He is the Founding Editor-in-Chief of the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE, the Technical Editor-in-Chief of *Memetic Computing*, and an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and the IEEE TRANSACTIONS ON CYBERNETICS.

Abhishek Gupta received the Bachelor of Technology degree from the National Institute of Technology Rourkela, Rourkela, India, in 2010, and the Ph.D. degree in engineering science from the University of Auckland, Auckland, New Zealand, in 2014.

He currently serves as a Scientist with the Singapore Institute of Manufacturing Technology, Agency for Science, Technology and Research, 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 current research interests include development of memetic computation as an approach for automated knowledge extraction and transfer across problems in evolutionary design.

Puay Siew Tan received the Ph.D. degree in context-aware B2B collaboration from Nanyang Technological University, Singapore, in 2010.

She leads the Manufacturing Control Tower as the Programme Manager which is responsible for the setup of Model Factory with the Singapore Institute of Manufacturing Technology, Singapore. Her research has been in the cross-field disciplines of computer science and operations research for cyber physical production system collaboration, in particular sustainable complex manufacturing and supply chain operations. She has been active in using context-aware and services techniques. Her current research interests include complex systems, specifically quantitative techniques for understanding disruptions propagation in networked supply chains and mitigation of risks caused by these disruptions.