Insights on Transfer Optimization: Because Experience is the Best Teacher

Article *in* IEEE Transactions on Emerging Topics in Computational Intelligence · November 2017

DOI: 10.1109/TETCI.2017.2769104

3 authors:

Abhishek Gupta
Singapore Institute of Manufacturing Technology (SIMTech)
95 PUBLICATIONS **1,745** CITATIONS
SEE PROFILE

Yew Soon Ong
Nanyang Technological University
397 PUBLICATIONS **13,237** CITATIONS
SEE PROFILE

Liang Feng
Chinese Academy of Sciences
262 PUBLICATIONS **10,213** CITATIONS
SEE PROFILE

Insights on Transfer Optimization: Because Experience is the Best Teacher

Abhishek Gupta, Yew-Soon Ong, and Liang Feng

***Abstract* – Traditional optimization solvers tend to start the search from scratch by assuming zero prior knowledge about the task at hand. Generally speaking, the capabilities of solvers do not automatically grow with experience. In contrast however, humans routinely make use of a pool of knowledge drawn from past experiences whenever faced with a new task. This is often an effective approach in practice as real-world problems seldom exist in isolation. Similarly, practically useful artificial systems are expected to face a large number of problems in their lifetime, many of which will either be repetitive or share domain-specific similarities. This view naturally motivates advanced optimizers that mimic human cognitive capabilities; leveraging on what has been seen before to accelerate the search towards optimal solutions of never before seen tasks. With this in mind, the present paper sheds light on recent research advances in the field of global black-box optimization that champion the theme of automatic knowledge transfer across problems. We introduce a general formalization of *transfer optimization*, based on which the conceptual realizations of the paradigm are classified into three distinct categories, namely, *sequential transfer*, *multitasking*, and *multiform optimization*. In addition, we carry out a survey of different methodological perspectives spanning Bayesian optimization and nature-inspired computational intelligence procedures for efficient encoding and transfer of knowledge building-blocks. Finally, real-world applications of the techniques are identified, demonstrating the future impact of optimization engines that evolve as better problem-solvers over time by learning from the past and from one another.**

***Index Terms* – Transfer, Multitasking, Multiform Optimization, Evolutionary Algorithms, Bayesian Optimization.**

I. Introduction

REAL-world problems seldom exist in isolation. As a result, humans routinely resort to various information sources, including a pool of knowledge extracted from past problem-solving experiences, when faced with a never before seen challenge or task. However, virtually all traditional optimization solvers, ranging from classical techniques to nature-inspired procedures, neglect this key aspect of human cognitive ability. In particular, a general shortcoming of many existing search strategies is that the optimization run typically begins from scratch, assuming a zero prior knowledge state. In many practical applications involving time sensitive actions and/or high cost of evaluations, ignoring the knowledge gained from previous optimization exercises can lead to deleterious computational overheads in the re-exploration of similar search spaces. Therefore, the ability to automatically transfer knowledge across problems is likely to have significant impact in dealing with the rapidly growing volume, variety, and complexity of the real-world problems of today.

Any practically useful system in an industrial setting is expected to tackle a large number of problems over a lifetime, many of which will either be repetitive or share domain-specific similarities. Thus, it is the ability to leverage on innate domain knowledge that often sets apart an expert from a novice. Notably, in machine learning, the idea of taking advantage of *available data* from related *sources* to improve the accuracy of the predictive function in a *target* task has received much interest under the label of *transfer learning* [1]-[3]. Nevertheless, associated research progress has largely been restricted to the domain of predictive analytics, where the availability of data makes it possible to ascertain the feasibility of knowledge transfer. For the case of black-box optimization, where little problem-specific data is available beforehand, efforts in automatic knowledge transfer have been relatively rare; thereby establishing the need to devise new *online* approaches that can harness recurrent patterns between problem-solving exercises. Preliminary efforts in this regard can be found in the evolutionary computation literature [4], where most approaches either rely on manual intervention to incorporate *a priori* heuristic knowledge into the search [5], or on the creation of an *artificial memory* providing a case-base (i.e., database) of past experiences [6]-[11]. However, in the latter, the elaborate case by case assessment required to yield relevant information was found to rapidly become prohibitive with the growing size of the database [12], [13].

In contrast to the above, humans can usually leverage enormous amounts of information gathered from experience, and effortlessly generalize the knowledge whenever faced with new tasks. The practical motivation for incorporating such cognitive capabilities into optimization solvers is derived from the growing presence of modern technologies such as cloud computing and the Internet of Things (IoT), which enable large-scale storage and seamless information communication facilities. In these settings, effectively capturing higher-order building-blocks of *generalizable* knowledge can play a significant role in enhancing the efficacy of problem-solving. To highlight this point further, consider the matter of representing the knowledge embedded in a large number (say N) of elite solutions, each comprising of B binary bits. If we naively store the raw data in memory,

Abhishek Gupta and Yew-Soon Ong are with the Data Science and Artificial Intelligence Research Centre (DSAIR), School of Computer Science and Engineering, Nanyang Technological University, Singapore. E-mail: {abhishekg, asysong}@ntu.edu.sg.

Liang Feng is with the College of Computer Science, Chongqing University, China. Email: liangf@cqu.edu.cn.

not only is the extracted knowledge too coarse (as the information required for a new problem may not be contained in the data), and possibly overfitting to the original problem(s) [13], [14], but also consumes $N{\cdot}B$ bits of memory. On the other hand, a simple computational model of the underlying probability distribution of the same solutions can represent potentially more generalizable higher-order knowledge while consuming only $O(B \cdot \log_2 N)$ bits of memory [15].

With the IoT giving rise to widespread inter-connectivity of physical devices and relatively easy access to diverse information streams, the present paper sheds light on the emerging scope of black-box optimization solvers to incorporate the general theme of *transfer optimization*. We present a formal motivation and definition of transfer optimization (in Section II) for it to serve as the common foundation for specific methodological offshoots. Based on our proposed definition, three distinct conceptual realizations of the paradigm are identified – namely, *sequential transfer* [16], *multitasking* [17], and *multiform optimization* [18] – that cumulatively encompass a range of ways in which transfer optimization can be put to use in practical settings.

Beyond formalizations, we draw attention to the most prominent algorithmic advances that have lately been achieved. Our survey spans Bayesian (in Section III) and nature-inspired computational intelligence techniques (in Section IV), which have emerged as independent tracks driving transfer optimization in practice [19]-[24]. The two methods originate from different philosophical perspectives, and have both attracted much interest in largely distinct domains. On one hand, Bayesian optimization is extremely data efficient, but is exclusively a model-based approach [19]. In contrast, nature-inspired techniques, albeit less data efficient, provide considerable flexibility with the interplay of evolutionary mechanisms and model-based transfer [16], [24]. With this in mind, the present paper attempts to provide a summary of the current state-of-the-art with a clear exposition of the complementary nature of different research strands, so as to facilitate a unification of ideas leading to the design of powerful transfer optimization engines in the future.

In order to emphasize the potential impact of successful transfer optimization, a diverse array of noteworthy real-world examples are identified, covering topics such as machine learning, robotics, engineering design, etc. (Section V). Thereafter, we outline promising future directions that are expected to play a pivotal role in establishing automatic knowledge encoding and transfer mechanisms as intrinsic features of optimization search (Section VI). Finally, Section VII encapsulates the paper and presents concluding remarks.

II. Preliminaries of Transfer Optimization

Consider a series of K optimization problems (or tasks) that are labeled as $\mathcal{T}_1$, $\mathcal{T}_2$, …, $\mathcal{T}_K$, belonging to domains $\mathcal{D}_1$, $\mathcal{D}_2$, …, $\mathcal{D}_K$, respectively. The k^{th} domain, denoted as $\mathcal{D}_k$, consists of a search space $\mathcal{X}_k$, and an auxiliary space $\mathcal{Y}_k$. To elaborate, $\mathcal{Y}_k$ contains the set Y_k of all possible operating conditions for which the optimization exercise may be carried out. For a particular instantiation of an optimization task $\mathcal{T}_k$ in $\mathcal{D}_k$, an element $\boldsymbol{y}_k$ of Y_k provides the specific operating conditions. Furthermore, $\mathcal{T}_k$ is described by an objective function f_k and a set of inequality and equality constraints $\boldsymbol{g}_k$ and $\boldsymbol{h}_k$, respectively. With this, the optimization problem formulation for $\mathcal{T}_k$ is stated as,

$$\begin{aligned} &\max_{\boldsymbol{x}} f_k(\boldsymbol{x}, \boldsymbol{y}_k), \\ &\text{subject to, } g_{ki}(\boldsymbol{x}, \boldsymbol{y}_k) \leq 0, \text{ for } i = 1, \dots, |\boldsymbol{g}_k|, \\ &\text{and, } h_{ki}(\boldsymbol{x}, \boldsymbol{y}_k) = 0, \text{ for } i = 1, \dots, |\boldsymbol{h}_k|. \end{aligned} \tag{1}$$

Here, f_k can either be a scalar, for a single-objective optimization problem (SOP), or a vector constituting a multi-objective optimization problem (MOP) – in which case it is written in boldface as $\boldsymbol{f}_k$. Further, $|\boldsymbol{g}_k|$ and $|\boldsymbol{h}_k|$ are the number of inequality and equality constraints, respectively. In Eq. (1), note that $\boldsymbol{y}_k \in Y_k$ is not directly part of the search, as we only optimize with respect to candidate solutions $\boldsymbol{x} \in \mathcal{X}_k$.

From a different point of view, when describing Eq. (1) in the context of search distributions instead of raw candidate solutions, its statement can be rewritten as,

$$\max_{p(\boldsymbol{x})} \int f_k(\boldsymbol{x}, \boldsymbol{y}_k) \cdot p(\boldsymbol{x}) \cdot \mathrm{d}\boldsymbol{x}.$$

Here, $p(\boldsymbol{x})$ represents the probability density function of candidate solutions in $\mathcal{X}_k$. Adhering to this probabilistic viewpoint, the operating conditions and the set of constraints of $\mathcal{T}_k$ induce a *prior* distribution $p_0(\boldsymbol{x} \mid \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k)$ over $\mathcal{X}_k$ at the onset of the search, such that,

$$p_0(\boldsymbol{x}) = 0 \text{ if } \boldsymbol{x} \notin X_k, \tag{2}$$

where $X_k \subseteq \mathcal{X}_k$ is the set of all seemingly admissible solutions of $\mathcal{T}_k$. Assuming little prior knowledge about the task, as is often the case for traditional black-box optimization algorithms, the prior distribution generally satisfies,

$$p_0(\boldsymbol{x}) > 0 \; \forall \boldsymbol{x} \; \in X_k, \tag{3}$$

which implies that the search assigns a positive sampling probability to all elements of the admissible set. A uniform prior is commonly used in this regard.

Next, consider V_k to be the set of all features spanned by the feature space $\mathcal{V}_k = \mathcal{X}_k \times \mathcal{Y}_k$ of domain $\mathcal{D}_k$. The dimensionality of $\mathcal{V}_k$ is the cardinality of the set V_k, which is denoted as $|V_k|$. Each constitutive feature of V_k imparts domain-specific contextual meaning that characterizes all optimization tasks within the domain. At a high-level, comparing the overlap in the domains (or feature spaces) of distinct tasks can provide qualitative hints on the suitability of knowledge transfer between them. Indeed, precise quantitative analysis of inter-task relationship must take into account $\{f, p_0(\boldsymbol{x} \mid \boldsymbol{y}, \boldsymbol{g}, \boldsymbol{h})\}$. However, in many real-world applications, analytical forms of the objective function and constraints may either be unavailable or inaccessible to rigorous mathematical treatment. Thus, in what follows, we categorize task pairs purely based on the extent of domain overlap, as a means of providing practical and intuitive guidelines to practitioners on the suitability of transfer optimization [21].

1) Complete domain overlap

For any two optimization tasks $\mathcal{T}_1$ and $\mathcal{T}_2$, their respective domains $\mathcal{D}_1$ and $\mathcal{D}_2$ are said to be completely overlapping if the features spanned by their corresponding features spaces are semantically the same, i.e., the relation $\mathcal{V}_1 = \mathcal{V}_2$ holds. Equivalently, denoting the intersection of feature sets as $V_{overlap} = V_1 \cap V_2$, we have,

$$V_1 \setminus V_{overlap} = \varnothing \quad \wedge V_2 \setminus V_{overlap} = \varnothing \quad . \tag{4}$$

2) Partial domain overlap

Domains $\mathcal{D}_1$ and $\mathcal{D}_2$ are said to be partially overlapping if there exists a subset of features that is unique to at least one task. This condition is expressed as follows,

$$V_{overlap} \neq \varnothing \quad \wedge (V_1 \setminus V_{overlap} \neq \varnothing \quad \vee V_2 \setminus V_{overlap} \neq \varnothing \quad). \tag{5}$$

3) No domain overlap

Finally, we label a pair of domains as being completely non-overlapping if,

$$V_{overlap} = \varnothing \quad . \tag{6}$$

Note that in all three aforementioned cases $|V_1|, |V_2| > 0$.

Ideally, with increasing values of $|V_{overlap}|/|V_1 \cup V_2|$, the efficacy of transfer optimization can be expected to grow in conjunction. However, even in cases of no domain overlap, the potential benefits of knowledge transfer cannot be immediately rejected. Indeed, some recent studies have empirically demonstrated that hidden correlations between tasks may be revealed through appropriate search space transformation schemes, such as domain adaptation [22] or cross-domain solution unification [23], [24].

Accordingly, in the formal setup that follows, we make the explicit assumption that a unification procedure exists that facilitates the alignment of features in a transformed space. Thus, a candidate solution $\boldsymbol{x}$ shall hereafter represent a point in such a unified space $\mathcal{X}$, from which it can be decoded to a task-specific solution in $\mathcal{X}_1$, or $\mathcal{X}_2$, …, or $\mathcal{X}_K$.

A. The Formalization

The key motivation of this work is to achieve human-level intelligence, particularly with regard to automatically learning from experience and generalizing the learned knowledge to solve related tasks more efficiently. To this end, we note that, *the increase in efficiency of a machine is traditionally defined as the increase in output of the machine per unit of input*. Even for the case of numerical algorithms, the same definition can be applied. In the context of optimization, the *output* can be interpreted as a *scalar* measure quantifying the quality Q of solution(s) obtained. On the other hand, the *input* is specified by available computational resources, such as the computing machinery and the time (or cost) budget. Accordingly, in subsequent formalizations, we denote the efficiency of a search algorithm on task $\mathcal{T}_k$ as $Q_t(\mathcal{T}_k)$, which represents the quality of solution(s) achieved with regard to f_k in 't' time-steps on a designated computer. In particular, if we denote the set of candidate solutions evaluated over 't' time-steps as X_k^t, then the algorithmic efficiency for an SOP can be stated as,

$$Q_t(\mathcal{T}_k) = f_k(\boldsymbol{x}^*) : \boldsymbol{x}^* \in X_k^t \wedge (\nexists \boldsymbol{x} \in X_k^t : f_k(\boldsymbol{x}) > f_k(\boldsymbol{x}^*)). \tag{7}$$

Even for the case of MOPs, where a Pareto optimal set of trade-off solutions are searched for [25], scalar efficiency measures can be specified based on commonly used quality indicators such as the hypervolume metric [26]. For the sake of brevity, we do not present details of the hypervolume or other related measures in this paper. However, it is important to highlight that given such a scalar measure, it is generally possible to analyze MOPs analogously to SOPs.

For a computational intelligence to learn with experience – i.e., for it to specifically demonstrate transfer optimization capabilities – it must be endowed with a knowledge base, which we denote as $\mathcal{M}$, for gathering information from different problem-solving exercises. Assuming the knowledge building-block extracted from $\mathcal{T}_k$ to be m_k, the knowledge base is considered to grow as,

$$\mathcal{M} = \cup_{\forall k} \, m_k. \tag{8}$$

Herein, we make an instinctive assumption that *the knowledge extracted a posteriori from an unknown optimization task is identical to the prior knowledge required to spontaneously address the same task*. With this, we interpret the effect of a knowledge building-block m_k as inducing a biased probability distribution $p_t(\boldsymbol{x} \mid f_k, \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k)$ that favors elite solutions of $\mathcal{T}_k$. We denote this relation as $m_k \rightarrow p_t(\boldsymbol{x} \mid f_k, \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k)$. Mathematically, the notion of the biased distribution is deemed to satisfy the following,

$$\int f_k(\boldsymbol{x}, \boldsymbol{y}_k) \cdot p_t(\boldsymbol{x} \mid f_k, \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k) \cdot \mathrm{d}\boldsymbol{x} \geq f_k^* - \varepsilon, \tag{9}$$

where (*) represents the global optimum, and ε (> 0) is a small convergence tolerance threshold. Based on the above, observe that if the prior in Eq. (3) is set as $p_0(\boldsymbol{x}) \leftarrow m_k$, then $\mathcal{T}_k$ will be spontaneously addressed, which aligns with our initial assumption about knowledge building-blocks. This implies that in scenarios where similar problems recur, solutions can be obtained faster by directly reusing one of $\{m_1, m_2, \ldots, m_{k-1}\}$ for $\mathcal{T}_k$. However, following Eq. (8), it can also be seen that,

$$\text{if } m_1 \approx m_2 \approx \ldots \approx m_K, \text{ then } \cup_{k \in \{1,2,..,K\}} \, m_k \approx m_1.$$

Clearly, $\mathcal{M}$ does not grow if only very similar problems are solved repeatedly. Thus, in order to continuously expand the knowledge base, it is crucial to tackle diverse optimization tasks. To elaborate, in a series of K tasks, $\mathcal{T}_K$ is said to be diverse relative to all other tasks if,

$$m_K \setminus \cup_{\forall k \neq K} \, m_k \neq \varnothing \quad . \tag{10}$$

Importantly, based on abstract probabilistic interpretations of knowledge, i.e., $m_k \rightarrow p_t(\boldsymbol{x} \mid f_k, \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k)$, the diversity of $\mathcal{T}_K$ may alternatively be stated as follows.

$$p_t(\boldsymbol{x} \mid f_K, \boldsymbol{y}_K, \boldsymbol{g}_K, \boldsymbol{h}_K) - \sum_{\forall k \neq K} \alpha_k \cdot p_t(\boldsymbol{x} \mid f_k, \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k) \neq 0,$$
$$\forall \boldsymbol{\alpha} = [\alpha_1;\ \alpha_2; \ldots] \text{ s.t. } \alpha_k \geq 0 \wedge \sum_{\forall k \neq K} \alpha_k = 1.$$

At this stage, note that even if distribution $p_t(\boldsymbol{x} \mid f_K, \boldsymbol{y}_K, \boldsymbol{g}_K, \boldsymbol{h}_K)$ cannot be precisely reconstructed using $\mathcal{M}$, the acquired knowledge base can still be useful for optimizing $\mathcal{T}_K$. Indeed, there may exist a latent vector $\boldsymbol{\alpha}^*$ of mixture coefficients for which the *gap* between $\sum_{\forall k \neq K} \alpha_k^* \cdot p_t(\boldsymbol{x} \mid f_k, \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k)$ and the *a priori* unknown distribution $p_t(\boldsymbol{x} \mid f_K, \boldsymbol{y}_K, \boldsymbol{g}_K, \boldsymbol{h}_K)$ is small (albeit non-zero). Therefore, assuming that an appropriate $\boldsymbol{\alpha} \approx \boldsymbol{\alpha}^*$ can be gleaned online while optimizing $\mathcal{T}_K$, relevant information can still be retrieved from $\mathcal{M}$ to accelerate the search.

In order to begin learning optimal mixture coefficients, the gap between distributions must first be quantified. In this regard, a commonly used measure with *convexity properties* is the Kullback-Leibler divergence (D_{KL}) [27]. In particular, D_{KL} specifies the amount of information lost when a distribution q is used to approximate distribution p;

$$D_{KL}(p||q) = \int p(\boldsymbol{x}) \cdot [\log p(\boldsymbol{x}) - \log q(\boldsymbol{x})] \cdot \mathrm{d}\boldsymbol{x}. \quad (11)$$

With this, the coefficient vector $\boldsymbol{\alpha}^*$ that minimizes the gap between $p_t(\boldsymbol{x} \mid f_K, \boldsymbol{y}_K, \boldsymbol{g}_K, \boldsymbol{h}_K)$ and $\sum_{\forall k \neq K} \alpha_k \cdot p_t(\boldsymbol{x} \mid f_k, \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k)$ is the optimal solution of the following mathematical program,

$$\min_{\boldsymbol{\alpha}} D_{KL}(p||q(\boldsymbol{\alpha})),$$
$$\text{where, } p = p_t(\boldsymbol{x} \mid f_K, \boldsymbol{y}_K, \boldsymbol{g}_K, \boldsymbol{h}_K), \quad (12)$$
$$\text{and } q(\boldsymbol{\alpha}) = \sum_{\forall k \neq K} \alpha_k \cdot p_t(\boldsymbol{x} \mid f_k, \boldsymbol{y}_k, \boldsymbol{g}_k, \boldsymbol{h}_k).$$

Eq. (12) sets out a blueprint for an *adaptive* transfer optimization algorithm in which the transfer of knowledge occurs by sampling solutions from the optimized mixture distribution. In particular, the coefficient α_k can be interpreted as a learned similarity measure between the k^{th} knowledge building-block and the current target task of interest, such that α_k determines the extent to which transfer occurs by setting the weight of the k^{th} probability distribution in the mixture.

Extending Eq. (12), if we consider $\bar{\boldsymbol{\alpha}} = [\boldsymbol{\alpha};\ \alpha_{add}]$, where α_{add} is the mixture coefficient corresponding to an additional knowledge building-block $m_{add} \rightarrow p_t(\boldsymbol{x} \mid f_{add}, \boldsymbol{y}_{add}, \boldsymbol{g}_{add}, \boldsymbol{h}_{add})$ extracted from task $\mathcal{T}_{add}$, then it follows that,

$$D_{KL}(p||q(\boldsymbol{\alpha}^*)) = \min_{\boldsymbol{\alpha}} D_{KL}(p||q([\boldsymbol{\alpha}; \alpha_{add} = 0]))$$
$$\geq \min_{\bar{\boldsymbol{\alpha}}} D_{KL}(p||q(\bar{\boldsymbol{\alpha}})). \quad (13)$$

Simply put, Eq. (13) indicates that additional problem-solving experiences should, in principle, monotonically enhance the ability to approach any desired target distribution arbitrarily closely. *Although such a target distribution is not known beforehand, it can be gradually approximated during the course of the search via known density estimation schemes*. Nevertheless, the key message of Eq. (13) is that, with a growing knowledge base $\mathcal{M}$, it is increasingly more plausible that the knowledge needed to solve a new task is in fact already contained in the knowledge base. With an *idealized* transfer optimization algorithm, it may be possible to glean the relevant knowledge online while automatically circumventing the deleterious effects of transferring useless (or possibly harmful) information (*negative transfer* [3]). Keeping this in mind, the principal goal of the transfer optimization paradigm is summarized by the following definition.

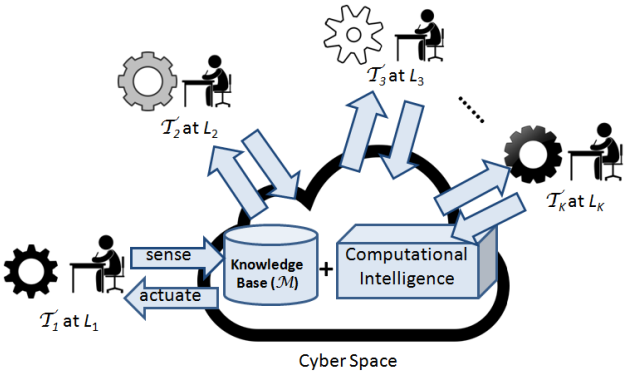


Fig. 1. The connectivity offered by IoT and cyber-physical systems, driven by a cloud infrastructure backbone, gives rise to the scope of seamless information communication across distinct problem-solving exercises at geographically distributed locations (L).

Definition (*Transfer Optimization*) Given a diverse experiential knowledge base $\mathcal{M} = \cup_{\forall k} m_k$, and a newly presented optimization task of interest ($\mathcal{T}$), *transfer optimization* facilitates performance speedup measured as $Q_t(\mathcal{T} \mid \mathcal{M}) - Q_t(\mathcal{T}) \geq 0$, where $Q_t(\mathcal{T} \mid \mathcal{M})$ is the algorithmic efficiency conditioned on the knowledge embedded in $\mathcal{M}$.

Notably, with the widespread inter-connectivity of physical devices offered by the IoT, the scope to build and leverage a rich knowledge base is greater than ever. This aspect is highlighted in Fig. 1, where the cyber space brings together geographically distributed physical systems, thereby making it possible for embedded solvers to harness large amounts of information shared by related tasks elsewhere. Similar ideas of automatic knowledge sharing apply to diverse applications such as multitasking robotics as well, with data streaming in through multiple sensory inputs at once.

As an aside, the definition above also sheds light on the impact of transfer optimization on the *inverse efficiency* of computational systems, suggesting that lesser compute power may be needed to achieve desired outputs. This view aligns with the recent impetus on moving computations closer to the edge of the IoT, such that devices with low computational capabilities can be directly utilized [28].

B. Categorizing Transfer Optimization

Our proposed definition for transfer optimization is quite broad, and gives rise to various conceptual realizations of the paradigm. In what follows, we classify these realizations into three distinct categories that shed light upon the range of ways in which transfer optimization can be put to use in practical settings.

1) Sequential Transfer

For sequential transfer optimization, we make the strict assumption that while tackling task $\mathcal{T}_K$, the tasks $\mathcal{T}_1$, $\mathcal{T}_2$, …, $\mathcal{T}_{K-1}$ have already been addressed previously with the extracted

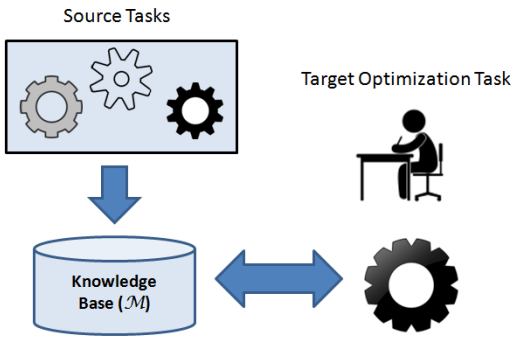


Fig. 2. In sequential transfer, the assumption is that while solving a new (target) optimization task, external information is made available from a knowledge base encompassing all tasks that have been tackled previously (labeled as source). Thus, the transfer is viewed as being largely unidirectional from the past to the present. For now, the fact that knowledge from the present can be used to refine what has been learned in the past is ignored for simplicity of exposition.

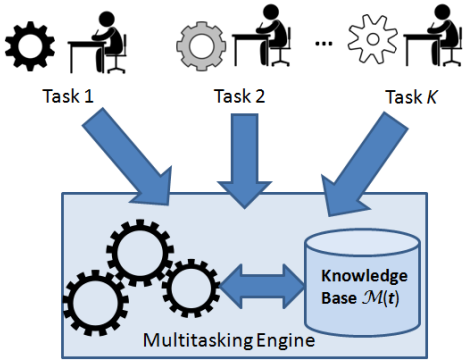


Fig. 3. During multitasking, multiple optimization tasks are tackled concurrently in a unified search space. The knowledge base $\mathcal{M}(t)$ is continuously updated and spontaneously shared among all tasks in the multitasking environment.

information available in the knowledge base $\mathcal{M}$. Herein, $\mathcal{T}_K$ is said to act as the target optimization task, while $\mathcal{T}_1$, $\mathcal{T}_2$, …, $\mathcal{T}_{K\text{-}1}$ are said to be source tasks – the situation is illustrated in Fig. 2. Thus, $Q_t^{transfer}(\mathcal{T}_K\,|\mathcal{M})$ represents the efficiency achievable on $\mathcal{T}_K$ conditioned on the knowledge captured in $\mathcal{M}$. Following the definition of transfer optimization, the aim for performance speedup as a consequence of sequential transfer is portrayed as,

$$Q_t^{transfer}(\mathcal{T}_K|\mathcal{M}) - Q_t(\mathcal{T}_K) \geq 0. \quad (14)$$

Here, $Q_t(\mathcal{T}_K)$ is the efficiency of a traditional optimization algorithm with no transfer, as given by Eq. (7).

The problem-solving efficacy of a computational system that successfully mimics human intelligence must ideally grow monotonically with experience (indicated by the size of $\mathcal{M}$). The viability of such an outcome – given an ideal transfer optimization algorithm – is reinforced by Eq. (13). With this, Eq. (14) may be further generalized;

$$Q_t^{transfer}(\mathcal{T}_K|\mathcal{M}) - Q_t^{transfer}(\mathcal{T}_K|\mathcal{M}') \geq 0 \text{ if } \mathcal{M}' \subseteq \mathcal{M}. \quad (15)$$

2) Multitasking

Different from sequential transfer, where we are concerned with optimizing a single target task at a time, multitasking caters to distinct tasks of equal priority occurring concurrently [24], [29]. Thus, in certain situations, it may not be possible to wait for one optimization task to be completed for knowledge to be made available for subsequent tasks. As an alternative, the optimization exercises $\mathcal{T}_1$, $\mathcal{T}_2$, …, $\mathcal{T}_K$ can progress in tandem, with the information generated being continuously updated and shared in the common knowledge base, which is immediately accessible to all tasks in the multitasking environment. A high-level schematic of multitasking is depicted in Fig. 3. Notice that as the knowledge base continuously evolves during the course of multitasking, it is denoted as a function of time $\mathcal{M}(t)$.

To emphasize, one of the major distinctions between sequential transfer and multitasking is that while the former is characterized by largely unidirectional transfer of knowledge from the past to the present, multitasking promotes *omnidirectional transfer* for more synergistic search.

Due to the simultaneous problem-solving, analyses of multitasking efficiency ($Q_t^{multitask}$) place requirements on the quality of solutions obtained across tasks, over a *cumulative time budget* of 't' time-steps, to be appropriately aggregated. Assuming this aggregation function to be Φ, we have,

$$Q_t^{multitask}(\mathcal{T}_1, \ldots, \mathcal{T}_K \,|\, \mathcal{M}(t)) = \Phi(Q(\mathcal{T}_1|\mathcal{M}(t)), \ldots, Q(\mathcal{T}_K|\mathcal{M}(t)). \quad (16)$$

where the efficiency achievable on each task is conditioned on $\mathcal{M}(t)$. The aggregation function is monotonic, which implies that for distinct algorithms $\mathcal{A}$ and $\mathcal{A}$', if measures Q and Q' follow $Q(\mathcal{T}_k) \geq Q'(\mathcal{T}_k)\ \forall k$, with at least one strict inequality, then $\Phi(Q(\mathcal{T}_1), \ldots, Q(\mathcal{T}_K)) > \Phi(Q'(\mathcal{T}_1), \ldots, Q'(\mathcal{T}_K))$. A sample aggregation technique has recently been reported in [30].

Given the same batch of K tasks, the efficiency of a traditional single-task optimization algorithm without the scope of knowledge transfer is simply,

$$Q_t(\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K) = \Phi(Q(\mathcal{T}_1), Q(\mathcal{T}_2), \ldots, Q(\mathcal{T}_K)). \quad (17)$$

Hence, the envisioned speedup due to multitasking suggests,

$$Q_t^{multitask}(\mathcal{T}_1, \ldots, \mathcal{T}_K \,|\, \mathcal{M}(t)) - Q_t(\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K) \geq 0. \quad (18)$$

3) Multiform Optimization

While sequential transfer and multitasking deal with distinct (self-contained) optimization tasks, multiform optimization is a novel concept for exploiting alternate formulations of a single target task of interest [18]. It is noted that in practical settings, several ways of formulating a particular optimization problem can be conceived, such as changing the structure of the objective function [31], [32], choosing the fidelity of an approximate objective function (in the spirit of multi-fidelity optimization [33]), deciding the number of control parameters needed [34], constrained/unconstrained formulations [35], etc.

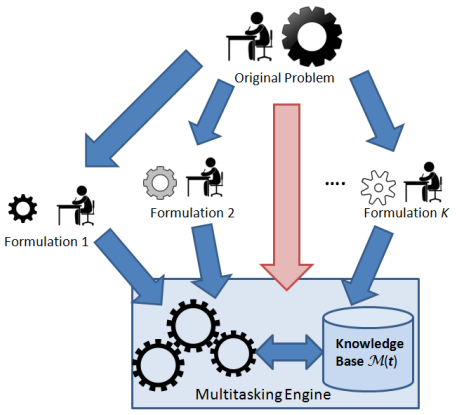


Fig. 4. Multiform optimization combines distinct formulations of a particular optimization task of interest into one all encompassing (multitasking) algorithm.

The challenge lies in the fact that it can often be difficult to ascertain which formulation is most suited for a particular problem at hand, given the known limits on computational resources. Indeed, different formulations induce different search behaviors, which may not be suitable in all scenarios. As illustrated in Fig. 4, in such cases, the basic idea of multiform optimization is to combine different formulations into a single all-encompassing (multitasking) algorithm, such that the hurdle of selecting a single formulation is bypassed. Most importantly, each formulation can serve as a *helper* (or *catalyst*) task [36], [37] in the multitasking environment, thereby allowing us to leverage the unique advantages offered by each of them through the process of continuous knowledge transfer.

As a simple illustration of an instantiation of multiform optimization, consider the notion of multiobjectivization [31]. To elaborate, in multiobjectivization, additional objectives are introduced in a manner such that, if $\boldsymbol{x}^*$ is an optimal solution of the original task $\mathcal{T}$, and X_R^* is the set of all Pareto optimal solutions of the reformulation $\mathcal{T}_R$, then,

$$\exists \boldsymbol{x}_R^* \in X_R^* : \boldsymbol{x}_R^* = \boldsymbol{x}^*. \tag{19}$$

It has been shown theoretically that multiobjectivization has the effect of introducing plateaus in the function landscape [38]. On one hand, this can have the positive effect of reducing local optima in the original formulation of the objective function. On the other hand, excessive plateaus may also make a problem more difficult to solve as an optimization algorithm is reduced to random walk behavior. This gives rise to a situation where multiform optimization can thrive, as shown in [18], as little can be said beforehand about which formulation is better suited for a particular problem instance.

With this, the conceived performance speedup through multiform optimization can be stated as,

$$Q_t^{multiform}(\mathcal{T} \mid \mathcal{T}_1, \ldots, \mathcal{T}_K, \mathcal{M}(t)) - Q_t(\mathcal{T}) \geq 0, \tag{20}$$

Where $\mathcal{T}$ is the original problem, $\mathcal{T}_1$, $\mathcal{T}_2$, …, $\mathcal{T}_K$ are alternate formulations, and $Q_t^{multiform}$ is the multiform efficiency.

III. Transfer Bayesian Optimization

Heretofore, we have laid down a basic structure for transfer optimization. In this section, we focus on a specific methodological perspective for knowledge transmission in practice – namely, *transfer Bayesian optimization*.

Bayesian optimization is a fundamentally model-based approach for tackling black-box problems characterized by high cost of function evaluations. Due to the considerable expense, there is strong emphasis on sample-efficiency. In other words, the knowledge embedded in solutions evaluated so far during an optimization exercise must be fully tapped while determining the most promising candidate solution to evaluate next. For any task $\mathcal{T}$, the technique iterates through the following steps: (a) learn a probabilistic model $p(f)$ – typically a Gaussian process – describing the objective function f, (b) use $p(f)$ to define a low cost acquisition function that provides a trade-off between exploration and exploitation while quantifying how promising it is to evaluate a particular point in the search space, and (c) optimize the acquisition function to determine the next point to evaluate using the expensive objective function [39], [40].

Although Bayesian optimization is highly sample-efficient compared to most other global optimization algorithms, it still requires tens to hundreds of evaluated solutions to build a sufficiently good model $p(f)$ that makes effective recommendations on the next solution to evaluate (also see Section IV-A). This is commonly referred to as the *cold start* problem, and has served as the main impetus to incorporate the notion of knowledge transfer in Bayesian optimization.

Recently, the majority of examples of practical Bayesian optimization with knowledge transfer across problems have been in the domain of automatic hyperparameter tuning of machine learning models [41]-[42], with certain methods reporting as much as 40% savings in optimization time (reduced from 10 days to 6 days) as opposed to the no transfer case [19]. Therefore, in the next subsection, we use this domain as the starting point for our subsequent discussions.

A. The Automatic Hyperparameter Tuning Problem

Considering $\boldsymbol{x}$ to denote hyperparameters, $\mathcal{A}$ to be the machine learning algorithm, and d to represent the dataset – which is split into a training set d_{train}, and a validation set d_{valid} on which the generalization error $f(\boldsymbol{x}, \mathcal{A}, d_{train}, d_{valid})$ of $\mathcal{A}$ is measured – the goal of hyperparameter optimization can be stated as follows,

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}, \mathcal{A}, d_{train}, d_{valid}). \tag{21}$$

As the evaluation of each candidate solution includes the training and validation of the machine learning algorithm, the tuning of hyperparameters can be extremely computationally expensive. The matter is further exacerbated in the case of big data, i.e., when the dataset d is very large.

To overcome the aforementioned challenge, it is contended that if d_1, d_2, …, d_{K-1} are different datasets on which $\mathcal{A}$ has been applied in the past, then the solutions/models generated during the previous hyperparameter optimization exercises may be useful when $\mathcal{A}$ is applied to a new dataset d_K. Indeed, it is this ability to harness experiential knowledge that

separates an expert (human) machine learning practitioner from a beginner. Importantly, the proposition fits perfectly within our conceived scope of transfer optimization, with a task $\mathcal{T}_k$ being associated to dataset d_k.

On comparing Eq. (21) with Eq. (1), d_k can be seen as resembling the operating conditions $\boldsymbol{y}_k$ for which optimization is to be carried out. Therefore, it is claimed that recently developed transfer Bayesian optimization algorithms for automatic hyperparameter tuning have immediate implications for general optimization problems as well. To emphasize the generality of our discussion, in the next subsection we replace dataset d with operating condition $\boldsymbol{y}$ throughout.

B. Methods of Transfer

In [19], the exchange of knowledge among tasks $\mathcal{T}_1$, $\mathcal{T}_2$, …, $\mathcal{T}_K$ was accomplished by using a multitask Gaussian process to learn a joint probabilistic model $p(f_1, f_2, \ldots, f_K)$. In particular, similarities across tasks were exploited by using the following product covariance function for solution and task pairs,

$$c((\boldsymbol{x}, \boldsymbol{y}_j),(\boldsymbol{x}', \boldsymbol{y}_k)) = c_X(\boldsymbol{x}, \boldsymbol{x}') \cdot c_{\mathcal{T}}(\boldsymbol{y}_j, \boldsymbol{y}_k), \tag{22}$$

where, $c_{\mathcal{T}}$ is the covariance between tasks and c_X is a correlation function between inputs [43]. The salient feature of the approach is that inter-task correlations are explicitly accounted for through $c_{\mathcal{T}}$. To elaborate, if two tasks $\mathcal{T}_j$ and $\mathcal{T}_k$ are indeed mutually informative, $c_{\mathcal{T}}(\boldsymbol{y}_j, \boldsymbol{y}_k)$ will assume a high magnitude while learning $p(f_1, f_2, \ldots, f_K)$. In contrast, if $\mathcal{T}_j$ and $\mathcal{T}_k$ are unrelated, then $c_{\mathcal{T}}(\boldsymbol{y}_j, \boldsymbol{y}_k) \approx 0$, so that the optimization exercise is not hampered due to harmful transfer.

Incidentally, accurately learning the parameters of the product covariance function becomes challenging when many tasks exist simultaneously. As an alternative, in [41], transfer was facilitated by constructing a common response surface for all tasks. A scenario was presented where optimization exercises for different operating conditions appear one after the other in a sequential manner. The authors assume that for similar tasks $\mathcal{T}_1$, $\mathcal{T}_2$, …, $\mathcal{T}_K$ the underlying functions $f_1, f_2, \ldots, f_K$ look qualitatively similar, although their location and the scale parameters can differ. Based on this assumption, when faced with $\mathcal{T}_K$, the common response surface is constructed by normalizing the objective function values as,

$$\tilde{f}_K = \frac{f_K - \mu}{\sigma}, \tag{23}$$

where μ and σ are the mean and standard deviation, respectively, calculated from the solutions evaluated so far for $\mathcal{T}_K$. The effect of the normalization procedure is to transform similar-looking functions in a manner such that they have comparable means and scale parameters as well, which allows a common Gaussian process model with shared covariance function to be transferred across tasks.

Along the same lines as the above, in [42] the common response surface was constructed by considering, for a given task $\mathcal{T}_k$, the ordinal (or ranking) information 'r' of the objective function f_k instead of its absolute value. To elaborate, $f_k(\boldsymbol{x}, \boldsymbol{y}_k) < f_k(\boldsymbol{x}', \boldsymbol{y}_k) \Leftrightarrow r(\boldsymbol{x}, \boldsymbol{y}_k) < r(\boldsymbol{x}', \boldsymbol{y}_k)$. Since rankings have a consistent scale, a single Gaussian process regression model $p(r)$ is built by combining the ranking information accumulated across all tasks. Thereafter, for a newly faced task $\mathcal{T}_K$, an acquisition function defined on $p(r)$ is optimized to determine the next candidate solution to evaluate.

While [19], [41], [42] put forward elaborate procedures for transfer to be carried out effectively, in [16] a simple meta-learning initialization scheme was proposed that can be incorporated within any traditional Bayesian optimization algorithm to combat the cold start problem. To this end, the initial solutions to be evaluated for building the probabilistic model $p(f_K)$ for a newly faced task $\mathcal{T}_K$ are biased towards the set of optimized solutions of tasks $\mathcal{T}_1$, $\mathcal{T}_2$, …, $\mathcal{T}_{K-1}$ solved in the past. Specifically, optimized solutions of those tasks that are most similar to $\mathcal{T}_K$ are considered. The similarity is determined by a learned regression model that captures the correlation $c(\boldsymbol{y}_j, \boldsymbol{y}_k)$ between any $\mathcal{T}_j$ and $\mathcal{T}_k$, where,

$$c(\boldsymbol{y}_j, \boldsymbol{y}_k) = r_s(f_j(\boldsymbol{x}, \boldsymbol{y}_j), f_k(\boldsymbol{x}, \boldsymbol{y}_k)). \tag{24}$$

In the above, r_s represents the Spearman rank correlation coefficient between the two functions.

IV. Transfer Evolutionary Optimization

In this section, we first compare and contrast Bayesian optimization with evolutionary algorithms (EAs). Thereafter, we discuss three different methodological perspectives that have been proposed in the literature for achieving knowledge transfer in evolutionary optimization.

A. Bayesian vs. Evolutionary Optimization

Despite many success stories, there are several challenges that limit the general applicability of Bayesian optimization. For instance, the methods described in Section III do not directly extend to combinatorial optimization problems as the covariance matrix of the probabilistic Gaussian process model becomes indefinite under combinatorial representations [44]. Further, handling high-dimensional search spaces in Bayesian optimization leads to considerable difficulties as the number of solution evaluations needed to get a good coverage of the search space increases exponentially. As a result, a probabilistic model built with a small number of evaluated solutions may no longer be sufficient for making effective prescriptions about the next point to evaluate.

It is worth mentioning that many of the difficulties associated with Bayesian optimization are commonplace in real-world applications. Therefore, a clear need arises for the development of alternate black-box optimization algorithms. EAs effectively fill this void. The key distinguishing feature of evolutionary methods is their fundamental reliance on evolutionary selection pressure (i.e., the principle of *survival of the fittest*) which generally acts on a *population* of search agents (or *individuals*). Typically, an individual solution in an EA is encoded as a chromosome (i.e., a string of genes) and endowed with a *fitness* that corresponds to its objective function value. If the fitness of the individual is relatively high (compared to the other individuals generated), its probability of survival and subsequent offspring creation is also high. On the other hand, if its fitness is low, it is bound to get gradually eliminated from the evolutionary search. The simplicity of the

idea provides much flexibility for the design of highly parallelizable EAs that can tackle various practically relevant scenarios, including high dimensional searches [45], and combinatorial representations [46]. Another feature of EAs is the seamless interplay possible between evolutionary mechanisms and model-based search. This has given rise to surrogate-assisted EAs, which form a class of methods tailored for computationally expensive problems; similar to Bayesian optimization. However, while Bayesian approaches require probabilistic function approximations, EAs can be combined with different surrogate models, including deterministic neural networks, polynomial response surfaces, etc. [47].

B. Genetic Transfer

The salient feature of EAs lends itself well to the transfer optimization paradigm. Specifically, if the knowledge transferred from a different optimization exercise is useful, the EA automatically preserves it and allows it to be further refined during the evolutionary search. However, if the transferred knowledge does not contribute to the solution of the current problem being solved, the selection pressure takes care of sieving out the useless (or harmful) genetic material.

In recent times, several attempts have been made to exploit the inherently adaptive nature of EAs for efficient transfer optimization. Given some form of cross-task solution unification (as alluded to in Section II), the general strategy is to bias the initial population distribution of a target task towards elite solutions obtained for source tasks through direct *seeding* of optimized genetic material [6], [48], [49]. The rationale behind such a strategy is that if the objective functions of tasks $\mathcal{T}_j$ and $\mathcal{T}_k$ are highly correlated in the ordinal sense, which implies,

$$f_j(\boldsymbol{x}) < f_j(\boldsymbol{x}') \Leftrightarrow f_k(\boldsymbol{x}) < f_k(\boldsymbol{x}'), \tag{25}$$

then optimizing one task immediately solves the other – by simply sharing the optimized genetic material. Contrarily, if $\mathcal{T}_j$ and $\mathcal{T}_k$ are uncorrelated, then the transferred genetic material is automatically ejected during the selection stage of the evolutionary search.

C. Evolutionary Multitasking

The motivation for evolutionary multitasking emerges from two observations that act over and above the ones discussed in the previous subsection. To begin, the implicit parallelism of a population offers an ideal platform for multiple concurrently occurring optimization tasks to be addressed without delay, such that the unified treatment enables latent correlations to be automatically harnessed during the search [21]. Further, for tasks that are not strongly correlated in the ordinal sense, it has been found that the (genetic) transfer of non-elite solutions often proves to be more useful [6]. However, this feature is not appropriately exploited by sequential transfer strategies where the initial population of the target task is simply biased towards elite source solutions. In contrast, algorithms for evolutionary multitasking facilitate the continuous exchange of genetic material throughout the course of the evolutionary search [24], thereby making it possible for all tasks in the multitasking environment to synergistically gain maximum benefits from one another.

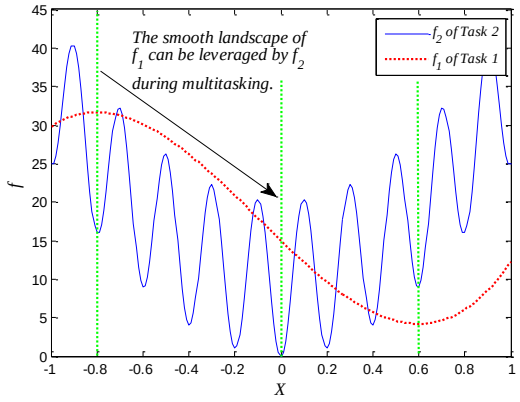


Fig. 5. A simple task can be exploited by a complex task in the same multitasking environment for faster convergence.

To further demonstrate this aspect of evolutionary multitasking, we refer to the illustrative minimization example in Fig. 5. Therein, task $\mathcal{T}_1$ is shown to have a relatively smooth objective function that is typically easier to optimize, while $\mathcal{T}_2$ possesses a rugged landscape such that any optimization algorithm has a tendency of getting trapped at a local optimum. The global optimum of $\mathcal{T}_2$ is located at 0 and that of $\mathcal{T}_1$ is located at 0.6. Interestingly, the point 0.6 is also a local optimum of $\mathcal{T}_2$ (as can be seen in Fig. 5). Assuming that $\mathcal{T}_1$ and $\mathcal{T}_2$ are solved in a sequential manner, if we bias the initial population distribution of $\mathcal{T}_2$ as $p_0(\boldsymbol{x}) \leftarrow p_t(\boldsymbol{x} \mid f_1)$, then the search is likely to stagnate at the local optimum. However, if $\mathcal{T}_1$ and $\mathcal{T}_2$ are solved together via multitasking, then, in the range [-0.8, 0], the continuous exchange of genetic material enables $\mathcal{T}_2$ to exploit the smooth landscape of f_1 as an alternate gradient descent direction, thereby avoiding several local optima to converge faster to its global optimum.

Nevertheless, as for transfer optimization in general, determining the suitability of evolutionary multitasking in arbitrary scenarios is not trivial. This is due to the lack of prior data in black-box optimization, which makes the viability of knowledge transfer difficult to ascertain beforehand. Thus, there arises the need for fast online measures of the similarity across problems – possibly based on the blueprint set out in Section II-A – such that adaptive multitasking can be carried out (as is currently possible in Bayesian optimization). We note that while offline methods to measure the synergy between optimization landscapes have been proposed for benchmark problems [50], these may rarely apply in practice where little is known beforehand about the objective function. Alternatively, the guidelines specified in Section II, based on the extent of overlap between the feature spaces of distinct tasks, provide useful intuition as to when multitasking may be successful. Further, examples in multiform optimization are deemed to be well suited for multitasking, as a fundamental relationship is already known to exist between the tasks.

D. Evolutionary Algorithms with Model-based Transfer

The techniques discussed heretofore were all essentially based on the direct transfer of genetic material across optimization tasks. However, as mentioned earlier, the flexibility offered by

EAs allows the interplay of evolutionary mechanisms with model-based transfer. To elaborate, the learned models serve to capture hidden patterns in optimized solutions, which may then be useful for future problem-solving exercises. In the memetic computing literature [51], such computational representations of knowledge are often referred to as *memes*, which can either be passed from one individual to another (via *imitation*) [52], or even be propagated across problems [16].

In contrast to genetic transfer, the hallmark of model-based transfer schemes is that the knowledge is represented in a more succinct manner (instead of storing the raw data), and shows lesser tendency of overfitting. Thus, while the success of genetic transfer is largely restricted to very similar problems with the same search space dimensionality, higher-order computational models can encapsulate generalizable knowledge with increased potential for transfer.

Lately, notable success stories have surfaced to substantiate the efficacy of EAs with model-based transfer. For instance, in [53] the model (m) takes the form of a positive semidefinite matrix that induces a modified *distance metric* for graph-based clustering cum sequencing problems – such as classical vehicle and arc routing [54]. In particular, given a pair of nodes n_a (with Cartesian coordinates $\boldsymbol{s}_a$) and n_b (with Cartesian coordinates $\boldsymbol{s}_b$) of a graph embedded in Euclidean space, the model scales the distance between them as,

$$dist(n_a, n_b) = \sqrt{(\boldsymbol{s}_a - \boldsymbol{s}_b)^T m (\boldsymbol{s}_a - \boldsymbol{s}_b)}, \tag{26}$$

so as to bring nodes that belong to the same cluster closer to one another, while moving nodes belonging to different clusters further apart. When faced with a new task, say $\mathcal{T}_K$, a combination of the previously learned models $m_1, m_2, \ldots, m_{K-1}$ is applied on the nodes of $\mathcal{T}_K$ to map to solutions that are (hopefully) close to the optimum clustering. Note that since the m's map to a small subset of all possible clusterings, they can be seen as inducing a biased distribution conditioned on the knowledge acquired from the past (which aligns with our abstract interpretation of m in Section II). It is worth mentioning that in [16], [55], the distance metric learning technique was extended to account for sequencing information as well. What is more, in [56], an *artificial neural network-based binary classification model* was proposed as a substitute to distance metric learning for identifying pairs of nodes that belonged to the same cluster.

Based on similar motivations as the above, in [57]-[59] it was shown that in estimation of distribution algorithms (which are a class of probabilistic model-based EAs [60]), recurrent patterns in the linkages between genes (variables) of discrete optimization problems could be gleaned from previous probabilistic models to guide the model learning process for related problems in the future. By doing so, not only was the complexity of model building reduced, but the search was also made more effective on future tasks. An alternate strategy within the same class of algorithms was proposed in [61], where, when a new problem arises, the probabilistic models of previously solved problems that are deemed similar are retrieved from a case base, combined [62], and then sampled to generate the initial population. The method is technically a lot like genetic transfer-based seeding.

Even in the domain of continuous optimization, promising results have been achieved by incorporating model-based transfer mechanisms. For example, in [63], a *decision tree* was learned on data generated during the optimization exercise to decipher the feasibility structure of an underlying highly constrained problem. Using the trained model to bias the initial population to lie entirely in the predicted feasible region resulted in substantial speedups of the evolutionary search on related problems in the future. More recently, in [64], a *denoising autoencoder* was used to learn a mapping between corresponding populations of distinct optimization tasks. To elaborate, at generation G, if the population of solutions (in matrix form) of the target task $\mathcal{T}_K$ is X_K^G, and the corresponding population of a previously solved source task $\mathcal{T}_k$ is X_k^G, then the denoising autoencoder (m_k) is learned as,

$$\min_{m_k} ||m_k \cdot X_k^G - X_K^G||_F, \tag{27}$$

where X_k^G is treated as a corrupted version of X_K^G. In Eq. (27), $||\cdot||_F$ indicates the Frobenius norm. The trained autoencoder was then used to transform and transfer the optimum solutions of the source task to the current target task of interest. Numerical experiments based on this approach demonstrated accelerated convergence characteristics for a variety of benchmark and real-world MOPs.

Finally, much like the recent advances seen in Bayesian optimization, there exists significant scope for augmenting surrogate-assisted EAs with concepts of adaptive transfer learning, such that data/models from related problems can be used to build accurate and low cost approximations of costly functions in the target task of interest. Preliminary work in this direction has been done for multi-fidelity problems [33], [65], where information from low fidelity models is used to accelerate high fidelity optimizations.

V. Practical Transfer Optimization Exemplars

Any practically useful system will generally face a large number of problems in its lifetime, most of which will either be repetitive or have domain-specific similarities. Mechanisms for automatically exploiting these latent similarities are the distinguishing feature of transfer optimization algorithms. In this section, we present latest research activities that offer a glimpse of the considerable utility of the paradigm across a wide array of real-world applications, ranging from machine learning to engineering design.

A. Transfer Optimization in Machine Learning

In addition to the automatic hyperparameter tuning problem discussed in Section III (which has largely been the arena of transfer Bayesian optimization), there exist a plethora of opportunities for the different conceptual realizations of transfer optimization to come to the fore in machine learning. Particularly noteworthy advancements have been made in transfer optimization enabled genetic programming (GP). As an example, [66]-[68] showed that by transferring optimized genetic material (in the form of trees or sub-trees of computer programs evolved by a GP solver) from a source to a target symbolic regression task helped achieve better training error as well as improved generalization performance. The authors

of [69], [70] applied a similar strategy to learning classifier systems, demonstrating that building-blocks of knowledge in the form of *code fragments* (GP-like sub-trees), that were extracted from small-scale problems, could be reused while learning more complex (large-scale) problems in the same or related domains akin to the behavior of human beings. In [71], it was revealed that this general approach can solve very difficult variants of the *n*-bit multiplexer problem that were previously insolvable by any other method.

The GP-based transfer scheme was further extended in [23] to the case of image classification. In particular, potentially useful code fragments extracted from simpler image classification problems were incorporated into the initial population and genetic mutation steps of a transfer learning GP algorithm, thereby leading to improved performance in complex, e.g., rotated and noisy, problems from similar as well as different domains.

While the examples presented above follow a procedure aligned to sequential transfer, there have also been recent applications of evolutionary multitasking in the field of machine learning. For instance, an evolutionary multitasking variant of Cartesian GP has been shown to solve a set of elementary logic functions twice as easily as with a direct single-task approach [72]. In [73], a multifactorial EA (MFEA) based multitasking engine [18], in conjunction with GP, was used for learning an ensemble of decision trees. The results showed that the multitasking algorithm was able to achieve classification accuracy comparable to an ensemble generated through multiple runs of traditional GP, and yet at the computational cost of only a single run. Likewise, in [74], a real-coded MFEA was used for modular training of feed-forward neural networks (where each task was described by a network of distinct width), showing superior convergence characteristics and classification performance in comparison to classical single-tasking for the *n*-bit parity problem. In [75], the basic idea of modular topologies through evolutionary multitasking was extended to extreme learning machines. Further, a co-evolutionary multitask learning approach has only recently been proposed for multi-step-ahead time series predictions [76], where different prediction horizons are seen as different but related tasks enabling transfer optimization.

B. Transfer Optimization in Robotics

A commonly encountered challenge in robot control tasks is the *bootstrap problem* [77], which concerns the lack of a sufficient fitness gradient during the initial stages of the search process. While evolving the controllers, the problem is further magnified when a randomly generated initial population is used. To overcome this hurdle, a genetic transfer-based *family bootstrapping* approach was proposed in [78], where the optimized solutions of a common source task were used to bias the initial population for associated target tasks. In other words, the optimized solutions from the source task formed the family ancestry for the subsequent stage of target tasks. The most interesting conclusion drawn from the experimental results was that creating the ancestry from a source MOP proved more helpful than a source SOP. This leads to the insight that the Pareto optimal solutions of an MOP can lead to a more diverse knowledge base, which, as has been argued in Section II-A, is beneficial for transfer optimization.

In addition to sequential knowledge transfer, the potential impact of evolutionary multitasking in robotics has been investigated in [21]. Therein, preliminary work was presented on improving the path planning of unmanned aerial vehicles by tackling multiple related missions simultaneously.

C. Transfer Optimization in Games

The computer gaming industry is rapidly growing, and has much benefitted from the use of artificial intelligence in enhancing game-play experience [79]. As a result, games are also commonly used by researchers as test beds for showcasing the efficacy of state-of-the-art algorithms [80]. It therefore comes as little surprise that the potential of transfer optimization has been established in computer games as well. In [81], EAs were used to develop player strategies that could lead to challenging opponents in tactical and strategic games. Specifically, the concept of genetic transfer was invoked to respond quickly to changing game dynamics – where a change was simply viewed as a new problem that is likely to share some similarities with previous settings. In addition to the above, knowledge was automatically acquired from human players by recording their game-play, so as to learn how to avoid potential traps.

The efficacy of an evolutionary transfer reinforcement learning framework for multi-agent systems, based on the concepts of memetic automaton [82], has lately been demonstrated on a first person shooter computer game [83]. The success of the approach is attributed to the scope of transfer across agents, which enables them to learn faster from better performing agents and thereby solve complex tasks more efficiently and effectively.

Besides computer games, the capability of solving complex puzzles can provide insights about the prowess of machine intelligence. In [84], Sudoku puzzles (which can be cast as highly constrained combinatorial optimization problems) were considered because of the interesting feature that outwardly unlike puzzles can often end up having final solutions that are alike; which provides an analogy to the prevalence of latent synergies between seemingly disparate problem-solving tasks. The numerical experiments showed that when latent synergies in the form of complementary constraints existed, the multitasking MFEA resulted in rapid streamlining of the search towards feasible solutions. However, it was also found that for the case of task clones (i.e., puzzles that were identical in every sense), multitasking performed the same as standard single-tasking; possibly because no new information was available from the other task. This led to the realization that in evolutionary multitasking, inter-task *complementarity* emerges from myriad interactions between tasks that may not be apparent to the eye. For the sake of brevity, we refrain from discussing this matter in depth in this paper. For preliminary thoughts on what it might mean for one task to complement another in a general multitask setting, the reader is referred to [50].

D. Transfer Optimization in Dynamic Environments

The ubiquitous need for prompt decision making in dynamic environments provides a perfect setting for transfer optimization. This is because any change in the environment can be seen to constitute a new problem, to which knowledge

can be transferred from previously tackled environments. When the changes are cyclic in time, the transfer of knowledge can work particularly well. These observations have been exploited for designing various approaches for optimization in dynamic environments, a review of which is available in [85]. For instance, an associative memory scheme was proposed in [86], where probabilistic models together with optimized solutions of past environments were stored for reuse whenever a change in the environment occurred. While [86] is memory-based, prediction-based approaches for dynamic environments have also been proposed in the past [87], where changes in the solutions are estimated based on changes that have occurred previously. More recently, the distance metric learning-based transfer procedure (described in Section IV-D) has also been adapted for handling dynamism in the logistics industry [88]. In particular, whenever new customer requests are received during the execution of a delivery plan, the knowledge captured from the previous time slot is applied for maximally aligning the customer distribution to the customer-vehicle assignments – thereby reducing the effort needed for re-optimization.

E. Transfer Optimization in Engineering Design

Engineering designs are usually gradually improved over time. Further, it is found that modern day design cycles are typically distributed in nature, consisting of multiple teams working on associated ideas in tandem [89]. Various conceptual designs are analyzed, before selecting the one that best suits a set of requirements [90]. Thus, manual knowledge adaptation and reuse is routine practice to speed up the process. By extension, the utility of transfer optimization emerges naturally, as the paradigm facilitates automatic exploitation of the overlaps between related designs.

A number of optimization strategies have been proposed lately for accelerating engineering design. In [64], the denoising autoencoder was used to map optimized solutions of previous process design exercises to the current target task of interest. Alternatively, a decision tree trained to decipher the feasibility structure of a problem was used in [63] as a transferrable nugget of knowledge within a family of processes for composite materials manufacturing.

Even the benefits of evolutionary multitasking have been demonstrated in the context of engineering design [91], [92]. To elaborate, by tackling designs of more-or-less similar type concurrently, the cost of exploring common parameter spaces was shown to be substantially reduced. This feature is strongly highlighted in [91] where multitasking was shown to push the envelope of EAs, facilitating simultaneous convergence to Pareto optimal solutions of multiple MOP design formulations at the same time. The efficacy of multitasking further extends to highly constrained search spaces common in engineering design, such that the distribution of feasible solutions is gleaned and automatically transferred from simpler to more difficult problems through continuous genetic exchange [93].

VI. Future Directions

So far in the paper, we have discussed two distinct research strands in the field of global black-box optimization that have addressed the notion of automatic knowledge transfer. To summarize, model-based approaches of Bayesian optimization algorithms is found to enable explicit learning of the similarity between optimization problems as the search progresses and data is gradually accumulated. As a result, the extent to which transfer should occur between tasks can be automatically modulated online during the course of the search. In other words, transfer Bayesian optimization provides an appealing option for truly adaptive transfer. However, there are certain shortcomings. To begin, the Gaussian process models that are typically used for probabilistic approximations of the objective function in Bayesian optimization do not directly extend to the case of combinatorial representations. Further, as the dimensionality of the problem grows, Bayesian optimization is severely hampered by the cold start problem, as an exponentially increasing number of data points are needed to start learning sufficiently informative models. Thus, despite its notable features, there exists a wide range of real-world scenarios in which Bayesian optimization may fail.

To deal with the aforementioned challenges, evolutionary computation has emerged as an attractive alternative. The flexibility offered by the simple mechanisms of EAs allows combinatorial representations, as well as problems with significantly larger search space dimensions, to be tackled with relative ease. Notably, EAs also allow for automatic knowledge incorporation through direct genetic transfer, or through different forms of model-based transfer. However, a drawback of existing methods in this regard is that they are generally found to be over reliant on the sieving effect of the evolutionary selection pressure. Sufficiently in-depth analysis is seldom done to infer the similarity across problems. Therefore, given many diverse information streams, it is highly likely that the selection pressure will be overburdened in the process of culling an abundance of useless information. In the worst case, this may severely hamper the overall efficacy of the evolutionary search process [94].

Taking note of the strengths and weaknesses of these two prominent strategies for black-box optimization, it is deemed that the future of transfer optimization as an industrial norm depends on a conceptual unification of the research strands. On one side, research efforts are needed towards the development of novel representation schemes and genetic operators that are better suited for seamless knowledge transfer and multitasking, such that the burden on the evolutionary selection module can be reduced. On the other side, lessons must be learned from the advances in transfer Bayesian optimization, such that some theoretical guarantees can be achieved with regard to minimizing the deleterious effects of harmful negative transfer through the explicit capture of the similarities across problems.

Given the path ahead of us, it is our view that the future lies in a new generation of memetic computing, where, with the widespread connectivity of devices supported by the IoT, the notion of memes in computational intelligence will take a form analogous to their social connotation. We elaborate in the next subsection.

A. The Emerging Problem-Solving Web

As has been alluded to throughout this paper, real-world problems seldom exist in isolation. The implications of this fact are further magnified with the dawn of the IoT. While optimization problems have largely been dealt with as self-

contained silos, the future shall give rise to complex networks of related problems, with each node in the network being either a self-contained task, or a single component of a much larger multicomponent problem. Memetic computing is expected to thrive in such settings. In particular, the interconnected *web* of problems will make it possible for knowledge memes generated at any node to propagate throughout the network (mimicking the viral effect [95]), such that other problem-solving exercises can immediately benefit from related experiences elsewhere.

Finally, we note that a conceptual simplification in the theoretical formalizations in this paper has been that a knowledge building-block (or meme) extracted from a particular task (or node) does not evolve despite what is learned from subsequent tasks. However, it is understood that as a meme propagates through a network of problems, it may be progressively augmented at each node, such that it houses increasingly complex knowledge that is suited for tasks of growing complexity. The utility of such progressive knowledge transfer for problem-solving has been shown in the context of neuroevolution of challenging controller design tasks [96]. Notably, the idea of continuously evolving memes is somewhat analogous to that of continual learning [97].

VII. Conclusions

This paper is dedicated to establishing the notion of *transfer optimization* as a novel paradigm facilitating the automatic transfer of knowledge across problems as a way to mimic an essential feature of human intelligence. A formalization of the paradigm was presented, showing, in particular, that with a growing knowledge base the problem-solving capability of an ideal transfer optimization algorithm will in principle grow in tandem. A rough blueprint for such an adaptive algorithm was also proposed based on the learning of an optimal mixture of knowledge represented in the form of probability density estimations. Following from the formalizations, three distinct conceptual realizations of transfer optimization were identified, namely, *sequential transfer* – where problems appear one after the other, *multitasking* – where multiple optimization tasks occur simultaneously, and *multiform optimization* – where multiple distinct formulations of a single target task of interest are tackled in conjunction.

In addition to introducing the general problem statement, various methodological perspectives spanning Bayesian as well as evolutionary techniques were surveyed, with the idea that the future of transfer optimization depends on a conceptual unification of the complementary aspects of the two research strands. Further, a variety of noteworthy real-world exemplars, ranging from machine learning to engineering design, were discussed, highlighting the practical implications of associated research activities.

To conclude, it is observed that the widespread connectivity offered by present-day technologies, such as IoT and cyber-physical systems, provides immense scope for harnessing the knowledge embedded in vast information streams from related tasks in geographically distributed locations. Such settings point towards a new generation of memetic computing, where knowledge memes extracted from a specific task can be spontaneously propagated through an inter-connected web of related problem-solving exercises, thereby lending memes in computational intelligence a form analogous to their social connotation.

Acknowledgement

This work is partially supported by the Data Science and Artificial Intelligence Research Centre (DSAIR) and the School of Computer Science and Engineering at Nanyang Technological University.

References

[1] Thrun, S. (1996, January). Is learning the n-th thing any easier than learning the first?. In *Advances in neural information processing systems* (pp. 640-646). MORGAN KAUFMANN PUBLISHERS.

[2] Caruana, R. (1998). Multitask learning. In *Learning to learn* (pp. 95-133). Springer US.

[3] Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345-1359.

[4] Jin, Y. (Ed.). (2013). *Knowledge incorporation in evolutionary computation* (Vol. 167). Springer.

[5] Becerra, R. L., & Coello, C. A. C. (2005). A cultural algorithm for solving the job shop scheduling problem. In *Knowledge Incorporation in Evolutionary Computation* (pp. 37-55). Springer Berlin Heidelberg.

[6] Louis, S. J., & McDonnell, J. (2004). Learning with case-injected genetic algorithms. *IEEE Transactions on Evolutionary Computation*, *8*(4), 316-328.

[7] Leake, D. B. (1996). *Case-Based Reasoning: Experiences, lessons and future directions*. MIT press.

[8] Kraay, D. R., & Harker, P. T. (1996). Case-based reasoning for repetitive combinatorial optimization problems, part I: Framework. *Journal of Heuristics*, *2*(1), 55-85.

[9] Bearpark, K., & Keane, A. J. (2005). The use of collective memory in genetic programming. In *Knowledge Incorporation in Evolutionary Computation* (pp. 15-36). Springer Berlin Heidelberg.

[10] Johnson, J., & Louis, S. J. (2005). Case-initialized genetic algorithms for knowledge extraction and incorporation. In *Knowledge incorporation in evolutionary computation* (pp. 57-79). Springer Berlin Heidelberg.

[11] Chang, P. C., Hsieh, J. C., & Wang, Y. W. (2005). Genetic algorithm and case-based reasoning applied in production scheduling. In *Knowledge Incorporation in Evolutionary Computation* (pp. 215-236). Springer Berlin Heidelberg.

[12] Meuth, R., Lim, M. H., Ong, Y. S., & Wunsch, D. C. (2009). A proposition on memes and meta-memes in computing for higher-order learning. *Memetic Computing*, *1*(2), 85-100.

[13] Salamó, M., & López-Sánchez, M. (2011). Adaptive case-based reasoning using retention and forgetting strategies. *Knowledge-Based Systems*, *24*(2), 230-247.

[14] Wills, L. M., & Kolodner, J. L. (1994, August). Towards more creative case-based design systems. In *AAAI* (Vol. 94, pp. 50-55).

[15] Sastry, K., Goldberg, D. E., & Llora, X. (2007, July). Towards billion-bit optimization via a parallel estimation of distribution algorithm. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation* (pp. 577-584). ACM.

[16] Feng, L., Ong, Y. S., Tan, A. H., & Tsang, I. W. (2015). Memes as building blocks: a case study on evolutionary optimization+ transfer learning for routing problems. *Memetic Computing*, *7*(3), 159-180.

[17] Gupta, A., Da, B., Yuan, Y., & Ong, Y. S. (2017). On the Emerging Notion of Evolutionary Multitasking: A Computational Analog of Cognitive Multitasking. In *Recent Advances in Evolutionary Multi-objective Optimization* (pp. 139-157). Springer International Publishing.

[18] Da, B., Gupta, A., Ong, Y. S., & Feng, L. (2016, July). Evolutionary multitasking across single and multi-objective formulations for improved problem solving. In *Evolutionary Computation (CEC), 2016 IEEE Congress on* (pp. 1695-1701). IEEE.

[19] Swersky, K., Snoek, J., & Adams, R. P. (2013). Multi-task Bayesian optimization. In *Advances in neural information processing systems* (pp. 2004-2012).

[20] Feurer, M., Springenberg, J. T., & Hutter, F. (2015, January). Initializing Bayesian Hyperparameter Optimization via Meta-Learning. In *AAAI* (pp. 1128-1135).

[21] Ong, Y. S., & Gupta, A. (2016). Evolutionary multitasking: a computer science view of cognitive multitasking. *Cognitive Computation*, *8*(2), 125-142.

[22] Bali, K. K., Gupta, A., Feng, L., Ong, Y. S., & Siew, T. P. (2017, June). Linearized domain adaptation in evolutionary multitasking. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (pp. 1295-1302). IEEE.

[23] Iqbal, M., Xue, B., Al-Sahaf, H., & Zhang, M. (2017). Cross-Domain Reuse of Extracted Knowledge in Genetic Programming for Image Classification. *IEEE Transactions on Evolutionary Computation*, *21*(4), 569-587.

[24] Gupta, A., Ong, Y. S., & Feng, L. (2016). Multifactorial evolution: toward evolutionary multitasking. *IEEE Transactions on Evolutionary Computation*, *20*(3), 343-357.

[25] Ehrgott, M. (2006). *Multicriteria optimization*. Springer Science & Business Media.

[26] Jiang, S., Ong, Y. S., Zhang, J., & Feng, L. (2014). Consistencies and contradictions of performance metrics in multiobjective optimization. *IEEE Transactions on Cybernetics*, *44*(12), 2391-2404.

[27] Joyce, J. M. (2011). Kullback-leibler divergence. In *International Encyclopedia of Statistical Science* (pp. 720-722). Springer Berlin Heidelberg.

[28] Yi, S., Li, C., & Li, Q. (2015, June). A survey of fog computing: concepts, applications and issues. In *Proceedings of the 2015 Workshop on Mobile Big Data* (pp. 37-42). ACM.

[29] Wen, Y. W., & Ting, C. K. (2017, June). Parting ways and reallocating resources in evolutionary multitasking. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (pp. 2404-2411). IEEE.

[30] Da, B., Ong, Y. S., Feng, L., Qin, A. K., Gupta, A., Zhu, Z., ... & Yao, X. (2016). Evolutionary Multitasking for Single-objective Continuous Optimization: Benchmark Problems, Performance Metric, and Baseline Results. *Nanyang Technological University, Singapore, Tech. Rep* University. http://www.cil.ntu.edu.sg/mfo/download.html.

[31] Handl, J., Lovell, S. C., & Knowles, J. (2008, September). Multiobjectivization by decomposition of scalar cost functions. In *International Conference on Parallel Problem Solving from Nature* (pp. 31-40). Springer Berlin Heidelberg.

[32] Yuan, Y., Ong, Y. S., Gupta, A., & Xu, H. (2017). Objective Reduction in Many-Objective Optimization: Evolutionary Multiobjective Approaches and Comprehensive Analysis. *IEEE Transactions on Evolutionary Computation*.

[33] Lim, D., Ong, Y. S., Jin, Y., & Sendhoff, B. (2008, September). Evolutionary optimization with dynamic fidelity computational models. In *International Conference on Intelligent Computing* (pp. 235-242). Springer, Berlin, Heidelberg.

[34] Kok, K. Y., & Rajendran, P. (2016). Differential-Evolution Control Parameter Optimization for Unmanned Aerial Vehicle Path Planning. *PloS one*, *11*(3), e0150558.

[35] Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer methods in applied mechanics and engineering*, *191*(11), 1245-1287.

[36] Caruana, R. (1995). Learning many related tasks at the same time with backpropagation. In *Advances in neural information processing systems* (pp. 657-664).

[37] Jin, Y., & Sendhoff, B. (1999). Knowledge incorporation into neural networks from fuzzy rules. *Neural Processing Letters*, *10*(3), 231-242.

[38] Brockhoff, D., Friedrich, T., Hebbinghaus, N., Klein, C., Neumann, F., & Zitzler, E. (2007, July). Do additional objectives make a problem harder?. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation* (pp. 765-772). ACM.

[39] Klein, A., Bartels, S., Falkner, S., Hennig, P., & Hutter, F. (2015, December). Towards efficient Bayesian optimization for big data. In *NIPS 2015 Bayesian Optimization Workshop*.

[40] Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148-175.

[41] Yogatama, D., & Mann, G. (2014). Efficient transfer learning method for automatic hyperparameter tuning. In *AISTATS* (pp. 1077-1085).

[42] Bardenet, R., Brendel, M., Kégl, B., & Sebag, M. (2013, June). Collaborative hyperparameter tuning. In *ICML (2)* (pp. 199-207).

[43] Bonilla, E. V., Chai, K. M., & Williams, C. (2008). Multi-task Gaussian process prediction. In *Advances in neural information processing systems* (pp. 153-160).

[44] Zaefferer, M., & Bartz-Beielstein, T. (2016, September). Efficient global optimization with indefinite kernels. In *International Conference on Parallel Problem Solving from Nature* (pp. 69-79). Springer International Publishing.

[45] Omidvar, M. N., Yang, M., Mei, Y., Li, X., & Yao, X. (2017). DG2: A Faster and More Accurate Differential Grouping for Large-Scale Black-Box Optimization. *IEEE Transactions on Evolutionary Computation*.

[46] Mühlenbein, H., Gorges-Schleuter, M., & Krämer, O. (1988). Evolution algorithms in combinatorial optimization. *Parallel Computing*, *7*(1), 65-85.

[47] Lim, D., Jin, Y., Ong, Y. S., & Sendhoff, B. (2010). Generalizing surrogate-assisted evolutionary computation. *IEEE Transactions on Evolutionary Computation*, *14*(3), 329-355.

[48] Koçer, B., & Arslan, A. (2010). Genetic transfer learning. *Expert Systems with Applications*, *37*(10), 6997-7002.

[49] Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S., & Zhang, G. (2015). Transfer learning using computational intelligence: a survey. *Knowledge-Based Systems*, *80*, 14-23.

[50] Gupta, A., Ong, Y. S., Da, B., Feng, L., & Handoko, S. D. (2016, July). Landscape synergy in evolutionary multitasking. In *Evolutionary Computation (CEC), 2016 IEEE Congress on* (pp. 3076-3083). IEEE.

[51] Chen, X., Ong, Y. S., Lim, M. H., & Tan, K. C. (2011). A multi-facet survey on memetic computation. *IEEE Transactions on Evolutionary Computation*, *15*(5), 591-607.

[52] Ong, Y. S., Lim, M. H., & Chen, X. (2010). Memetic computation—past, present & future [research frontier]. *IEEE Computational Intelligence Magazine*, *5*(2), 24-31.

[53] Feng, L., Ong, Y. S., Tsang, I. W. H., & Tan, A. H. (2012, June). An evolutionary search paradigm that learns with past experiences. In *Evolutionary Computation (CEC), 2012 IEEE Congress on* (pp. 1-8). IEEE.

[54] Toth, P., & Vigo, D. (Eds.). (2014). *Vehicle routing: problems, methods, and applications*. Society for Industrial and Applied Mathematics.

[55] Feng, L., Ong, Y. S., Lim, M. H., & Tsang, I. W. (2015). Memetic search with interdomain learning: A realization between CVRP and CARP. *IEEE Transactions on Evolutionary Computation*, *19*(5), 644-658.

[56] Feng, L., Ong, Y. S., & Lim, M. H. (2013). Extreme Learning Machine Guided Memetic Computation for Vehicle Routing. *IEEE Intelligent Systems*, *28*(*6*), 38-41.

[57] Pelikan, M., Hauschild, M., & Lanzi, P. (2012). Transfer learning, soft distance-based bias, and the hierarchical boa. *Parallel Problem Solving from Nature-PPSN XII*, 173-183.

[58] Hauschild, M. W., Pelikan, M., Sastry, K., & Goldberg, D. E. (2012). Using previous models to bias structural learning in the hierarchical BOA. *Evolutionary Computation*, *20*(1), 135-160.

[59] Hauschild, M. W., & Pelikan, M. (2009, July). Intelligent bias of network structures in the hierarchical BOA. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation* (pp. 413-420). ACM.

[60] Pelikan, M. (2005). Hierarchical Bayesian optimization algorithm. *Hierarchical Bayesian Optimization Algorithm*, 105-129.

[61] Kaedi, M., & Ghasem-Aghaee, N. (2011). Biasing Bayesian optimization algorithm using case based reasoning. *Knowledge-Based Systems*, *24*(8), 1245-1253.

[62] Jiang, C. A., Leong, T. Y., & Kim-Leng, P. O. H. (2005). PGMC: a framework for probabilistic graphical model combination. In *AMIA Annual Symposium Proceedings* (Vol. 2005, p. 370). American Medical Informatics Association.

[63] Lim, D., Ong, Y. S., Gupta, A., Goh, C. K., & Dutta, P. S. (2016). Towards a new Praxis in optinformatics targeting knowledge re-use in evolutionary computation: simultaneous problem learning and optimization. *Evolutionary Intelligence*, *9*(4), 203-220.

[64] Feng, L., Ong, Y. S., Jiang, S., & Gupta, A. (2017). Autoencoding Evolutionary Search with Learning across Heterogeneous Problems. *IEEE Transactions on Evolutionary Computation*, *21*(5), 760-772.

[65] Forrester, A. I., Sóbester, A., & Keane, A. J. (2007, December). Multi-fidelity optimization via surrogate modelling. In *Proceedings of the royal society of london a: mathematical, physical and engineering sciences* (Vol. 463, No. 2088, pp. 3251-3269). The Royal Society.

[66] Dinh, T. T. H., Chu, T. H., & Nguyen, Q. U. (2015, May). Transfer learning in genetic programming. In *Evolutionary Computation (CEC), 2015 IEEE Congress on* (pp. 1145-1151). IEEE.

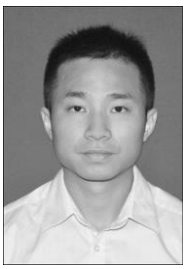
[67] Haslam, E., Xue, B., & Zhang, M. (2016, July). Further investigation on genetic programming with transfer learning for symbolic regression. In *Evolutionary Computation (CEC), 2016 IEEE Congress on* (pp. 3598-3605). IEEE.

[68] O'Neill, D., Al-Sahaf, H., Xue, B., & Zhang, M. (2017, June). Common subtrees in related problems: A novel transfer learning approach for

genetic programming. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (pp. 1287-1294). IEEE.
[69] Iqbal, M., Browne, W. N., & Zhang, M. (2014). Reusing building blocks of extracted knowledge to solve complex, large-scale boolean problems. *IEEE Transactions on Evolutionary Computation*, *18*(4), 465-480.
[70] Iqbal, M., Browne, W. N., & Zhang, M. (2012, July). Extracting and using building blocks of knowledge in learning classifier systems. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation* (pp. 863-870). ACM.
[71] Alvarez, I. M., Browne, W. N., & Zhang, M. (2016, July). Human-inspired Scaling in Learning Classifier Systems: Case Study on the n-bit Multiplexer Problem Set. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference* (pp. 429-436). ACM.
[72] Scott, E. O., & De Jong, K. A. (2017). Multitask Evolution with Cartesian Genetic Programming. *arXiv preprint arXiv:1702.02217*.
[73] Wen, Y. W., & Ting, C. K. (2016, July). Learning ensemble of decision trees through multifactorial genetic programming. In *Evolutionary Computation (CEC), 2016 IEEE Congress on* (pp. 5293-5300). IEEE.
[74] Chandra, R., Gupta, A., Ong, Y. S., & Goh, C. K. (2017). Evolutionary Multi-task Learning for Modular Knowledge Representation in Neural Networks. *Neural Processing Letters*, 1-17.
[75] Tang, Z., Gong, M., & Zhang, M. (2017, June). Evolutionary multi-task learning for modular extremal learning machine. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (pp. 474-479). IEEE.
[76] Chandra, R., Ong, Y. S., & Goh, C. K. (2017). Co-evolutionary multi-task learning with predictive recurrence for multi-step chaotic time series prediction. *Neurocomputing*, *243*, 21-34.
[77] Israel, S., & Moshaiov, A. (2012). Bootstrapping aggregate fitness selection with evolutionary multi-objective optimization. *Parallel Problem Solving from Nature-PPSN XII*, 52-61.
[78] Moshaiov, A., & Tal, A. (2014, July). Family bootstrapping: A genetic transfer learning approach for onsetting the evolution for a set of related robotic tasks. In *Evolutionary Computation (CEC), 2014 IEEE Congress on* (pp. 2801-2808). IEEE.
[79] Millington, I., & Funge, J. (2016). *Artificial intelligence for games*. CRC Press.
[80] Salimans, T., Ho, J., Chen, X., & Sutskever, I. (2017). Evolution Strategies as a Scalable Alternative to Reinforcement Learning. *arXiv preprint arXiv:1703.03864*.
[81] Louis, S. J., & Miles, C. (2005). Playing to learn: Case-injected genetic algorithms for learning to play computer games. *IEEE Transactions on Evolutionary Computation*, *9*(6), 669-681.
[82] Zeng, Y., Chen, X., Ong, Y. S., Tang, J., & Xiang, Y. (2017). Structured Memetic Automation for Online Human-Like Social Behavior Learning. *IEEE Transactions on Evolutionary Computation*, *21*(1), 102-115.
[83] Hou, Y., Ong, Y. S., Feng, L., & Zurada, J. M. (2017). An Evolutionary Transfer Reinforcement Learning Framework for Multi-Agent System. *IEEE Transactions on Evolutionary Computation*, *21*(4), 601-615.
[84] Gupta, A., & Ong, Y. S. (2016, December). Genetic transfer or population diversification? deciphering the secret ingredients of evolutionary multitask optimization. In *Computational Intelligence (SSCI), 2016 IEEE Symposium Series on* (pp. 1-7). IEEE.
[85] Yang, S. (2008). Genetic algorithms with memory-and elitism-based immigrants in dynamic environments. *Evolutionary Computation*, *16*(3), 385-416.
[86] Yang, S., & Yao, X. (2008). Population-based incremental learning with associative memory for dynamic environments. *IEEE Transactions on Evolutionary Computation*, *12*(5), 542-561.
[87] Zhou, A., Jin, Y., Zhang, Q., Sendhoff, B., & Tsang, E. (2007, March). Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization. In *International Conference on Evolutionary Multi-Criterion Optimization* (pp. 832-846). Springer, Berlin, Heidelberg.
[88] Zhou, L., Feng, L., Gupta, A., Ong, Y. S., Liu, K., Chen, C., ... & Yan, B. W. (2017, June). Solving dynamic vehicle routing problem via evolutionary search with learning capability. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (pp. 890-896). IEEE.
[89] Min, A. T. W., Sagarna, R., Gupta, A., Ong, Y. S., & Goh, C. K. (2017). Knowledge Transfer Through Machine Learning in Aircraft Design. *IEEE Computational Intelligence Magazine*, *12*(4), 48-60.
[90] Avigad, G., & Moshaiov, A. (2009). Set-based concept selection in multi-objective problems: optimality versus variability approach. *Journal of Engineering Design*, *20*(3), 217-242.
[91] Gupta, A., Ong, Y. S., Feng, L., & Tan, K. C. (2016). Multiobjective multifactorial optimization in evolutionary multitasking. *IEEE transactions on cybernetics*, *47*(7), 1652-1665.
[92] Gupta, A., Mańdziuk, J., & Ong, Y. S. (2015). Evolutionary multitasking in bi-level optimization. *Complex & Intelligent Systems*, *1*(1-4), 83-95.
[93] Cheng, M. Y., Gupta, A., Ong, Y. S., & Ni, Z. W. (2017). Coevolutionary multitasking for concurrent global optimization: With case studies in complex engineering design. *Engineering Applications of Artificial Intelligence*, *64*, 13-24.
[94] Liaw, R. T., & Ting, C. K. (2017, June). Evolutionary many-tasking based on biocoenosis through symbiosis: A framework and benchmark problems. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (pp. 2266-2273). IEEE.
[95] Weng, L., Menczer, F., & Ahn, Y. Y. (2013). Virality prediction and community structure in social networks. *Scientific reports*, *3*, 2522.
[96] Gomez, F. J., & Miikkulainen, R. (1999, July). Solving non-Markovian control tasks with neuroevolution. In *IJCAI* (Vol. 99, pp. 1356-1361).
[97] Ring, M. B. (1997). CHILD: A first step towards continual learning. *Machine Learning*, *28*(1), 77-104.

Abhishek GUPTA received the PhD degree in Engineering Science from the University of Auckland, New Zealand, in 2014. He is currently a Research Scientist at the School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore. Abhishek has diverse research experiences in computational science, ranging from the numerical modelling of solids and fluids, to topics in computational intelligence. Currently, his main research interests lie in the development of memetic computing as an approach for automatic learning and transfer of knowledge across optimization problems, with applications in design.

Yew-Soon ONG received a PhD degree on Artificial Intelligence in complex design from the Computational Engineering and Design Center, University of Southampton, UK in 2003. He is a Professor and the Chair of 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 Principal Investigator of the Data Analytics and Complex Systems Programme at the Rolls-Royce@NTU Corporate Lab. His research interest in computational intelligence spans across memetic computing, complex design optimization, and big data analytics. He is the founding Editor-in-Chief of the IEEE Transactions on Emerging Topics in Computational Intelligence, Associate Editor of the IEEE Transactions on Evolutionary Computation, the IEEE Transactions on Neural Networks & Learning Systems, the IEEE Transactions on Cybernetics, and others.



Liang FENG received the PhD degree from the School of Computer Engineering, Nanyang Technological University (NTU), Singapore, in 2014. He was a Postdoctoral Research Fellow at the Computational Intelligence Graduate Lab at NTU Singapore. He is currently an Assistant Professor with the College of Computer Science, Chongqing University, China. His research interests include Computational and Artificial Intelligence, Memetic Computing, Big Data Optimization and Learning, as well as Transfer Learning.