A Probabilistic Memetic Framework

(Quang Huy Nguyen, Yew-Soon Ong and Meng Hiot Lim)*
nguy0046@ntu.edu.sg, asysong@ntu.edu.sg, emhlim@ntu.edu.sg

ABSTRACT

Memetic algorithms (MAs) represent one of the recent growing areas in evolutionary algorithm (EA) research. The term MAs is now widely used as a synergy of evolutionary or any population-based approach with separate individual learning or local improvement procedures for problem search. Quite often, MAs are also referred to in the literature as Baldwinian EAs, Lamarckian EAs, cultural algorithms or genetic local search. In the last decade, MAs have been demonstrated to converge to high quality solutions more efficiently than their conventional counterparts on a wide range of real world problems. Despite the success and surge in interests on MAs, many of the successful MAs reported have been crafted to suit problems in very specific domains.

Given the restricted theoretical knowledge available in the field of MA and the limited progress made on formal MA frameworks, we present a novel **Pr**obabilistic **M**emetic **F**ramework that models MA as a process involving the decision of embracing the separate actions of evolution or individual learning and analyze the probability of each process in locating the global optimum. Further, the framework balances evolution and individual learning by governing the learning intensity of each individual according to the theoretical upper bound derived while the search progresses.

Theoretical and empirical studies on representative benchmark problems commonly used in the literature are presented to demonstrate the characteristics and efficacies of the probabilistic memetic framework. Further, comparisons to recent state-of-the-art evolutionary algorithms, memetic algorithms and hybrid evolutionary-local search demonstrate that the proposed framework yields robust and improved search performance.

Keywords: Memetic algorithm (MA), hybrid genetic algorithm-local search (GA-LS), probabilistic evolutionary algorithms

* Manuscript received ; Q. H. Nguyen and Y. S. Ong are with the School of Computer Engineering while M. H. Lim is with the School of Electrical and Electronic Engineering, Nanyang Technological University, 639798, Singapore.

1 Introduction

Optimization may be defined simply as a process for seeking better or best alternative solution from a number of possible solutions to a problem. It is part and parcel of problem-solving in many areas of science and engineering, including those that are directly applicable in our daily life. Over the years, optimization methods have evolved considerably, with many algorithms and implementations now available. Typical conventional search methods are steepest-descent methods, conjugate-gradient, quadratic programming, and linear approximation methods. These local strategies carry out iterative search for the optimum solution within the neighborhood of a candidate solution. Starting from a single candidate solution, these methods rely on "local" information to decide on their next move in the neighborhood. As the name suggests, all deterministic local strategies suffer from the same drawback; they are susceptible to stagnation due to convergence towards a local optima. Their main advantage is the efficiency. However, they tend to be sensitive to starting point selection and are more likely to settle at non-global optima than modern stochastic algorithms.

Modern stochastic algorithms such as evolutionary algorithms (EA) draw inspiration from biological evolution. EA, unlike conventional numerical optimization methods, produce new design points that do not use information about the local slope of the objective function and are thus not prone to stalling at local optima. Instead, they involve a search from a "population" of solutions, making use of competitive selection, recombination, mutation or other stochastic operators to generate new solutions which are biased towards better regions of the search space. Furthermore, they have shown considerable success in dealing with optimization problems characterized by non-convex, disjoint or noisy solution spaces. Some of the modern stochastic optimizers that have attracted a great deal of attention in recent years include simulated annealing, tabu search, genetic algorithms, evolutionary programming, evolution strategies, differential evolution and others [1]-[5]. These stochastic methods have been successfully applied to many real world optimization problems.

In general, evolutionary algorithms are capable of exploring and exploiting promising regions of the search space. They can, however, take a relatively long time to locate the exact local optimum within the region of convergence. Memetic algorithms (MAs) are recent extensions of evolutionary algorithms with the introduction of individual learning as a separate process of local refinement for accelerating search. Recent studies on MAs have demonstrated that they converge to high quality solutions more efficiently than their conventional counterparts [6]-[11] on many real world applications. In recent years, many dedicated MAs have been crafted to solve domain-specific problems more efficiently. Meanwhile, a distinct group of researchers concentrated on the algorithmic aspects of MA, as combinations of EAs with local searches [13]-[24]. In recent special issues [25], [26], and journal [53] dedicated to MA research, several new design methodologies of memetic algorithms [27]-[29], [54], [55], and specialized memetic algorithms designed for tackling the permutation flow shop scheduling [56], optimal control systems of permanent magnet synchronous motor [28], VLSI floorplanning [30], quadratic assignment problem [31],

gene/feature selection [31], have been introduced. From a survey of the area, it is now well established that potential algorithmic improvement can be achieved by considering some important issues of MA:

1) How often should local learning be applied, i.e., local search frequency, f_{il}? Alternatively, what is the appropriate probability P_{il} for applying local learning to an individual?
2) To which solution subset of the population, Ω_{il}, should the local learning be applied?
3) How long should the local learning be run, i.e., local search intensity or local search computational time budget, t_{il}?
4) Which local improvement procedure or meme to use?

One of the first issues pertinent to memetic algorithm design is to consider how often the local search should be applied, i.e., local search frequency. In [13], the effect of local search frequency on MA search performance was considered where various configurations of the local search frequency at different stages of the MA search were investigated. Conversely, it was shown in [14] that it may be worthwhile to apply local search on every individual if the computational complexity of the local search is relatively low. On the issue of selecting appropriate individuals among the EA population that should undergo local search, fitness-based and distribution-based strategies were studied for adapting the probability of applying local search on the population of chromosomes in continuous parametric search problems with Land [15] extending the work to combinatorial optimization problems. Recently, Bambha et al. [16] introduced a simulated heating technique for systematically integrating parameterized local search into evolutionary algorithms to achieve maximum solution quality. Goldberg and Voessner also attempted to optimize the performance of global-local search hybrid in the context of a fixed local search time that leads to solution with acceptable targets on simplistic synthetic functions [17]. Subsequently, Sinha et. al. [18] extends the work in [16] with some in-depth study on genetic algorithm as a global searcher. The performance of MA search is also greatly affected by the choice of neighborhood structures. The effect of neighborhood structures on evolutionary search performances was first studied by Yao in the context of simulated annealing [19]. Recently, Krasnogor [20], [21] investigated how to change the size and the type of neighborhood structures dynamically in the framework of multi-meme memetic algorithms where each meme biases different neighborhood structure, acceptance rule and local search intensity. Various schemes for choosing among multiple local learning procedures or memes during the memetic algorithm search in the spirit of Lamarckian learning, otherwise, known as meta-Lamarckian learning was also addressed in Ong et. al. [6]. For a detailed taxonomy and comparative study on adaptive choice of memes in MA, the reader is referred to [22]. Based on the terminology provided in [22], several new adaptive and self-adaptive memetic algorithms were proposed by Smith **Error! Reference source not found.**, Liu et. al. **Error! Reference source not found.** and Caponio et. al. [28] and reported in the recent special issue on MA [25].

Thus far, it is clear that unless one has a priori knowledge as to whether the evolutionary or individual learning mechanisms suits the problem in hand, i.e., through the correct definition of the algorithmic settings, the MA may not perform at its optimum or worse, it may perform poorer than evolution or

individual learning alone. Depending on the complexity of the design problem, an algorithmic configuration that may have proven to be successful in the past might not work as well, or at all, on others - an outcome that is directly related to the "no free lunch theorem for search" [33]. With the restricted amount of theory currently available for choosing the appropriate configuration of MA that best matches a black box problem, it is not surprising that researchers have now opted for algorithms that learn and reconfigure itself to adapt to the problem in hand while the search progresses [13] - [24].

In the context of optimization, it is worth noting that the key design issue of MA lies in the successful promotion of competition and cooperation between the forces of *evolution* and *individual learning*. This can be achieved by adapting the algorithmic parameters of MA, within some limited computational budget. Since MA are synergies of *evolution* and *individual learning*, it may be worthwhile to model MA as a process involving the decision of embracing the separate actions of *evolution* or *individual learning* at each point in time. As an example, it is worth noting that the frequency and intensity of individual learning, i.e., f_{il} or t_{il}, are among those that directly define the degree of evolution (exploration) against individual learning (exploitation) in the MA search under some limited computational budget. Clearly, a more intense individual learning, i.e., a large t_{il} and f_{il}, provide greater chances of convergence to the local optima but limits the extent of evolution that may be expended without incurring excessive computational resources.

In this paper, we present a formal **Pr**obabilistic **M**emetic **F**ramework (PrMF) that governs at runtime whether evolution or individual learning should be favored, with the primary objective of accelerating MA search. Note that existing strategies for controlling the MA parameters are mostly designed based on heuristics that comes with little theoretical motivation and considers each design issue of MA independently. In contrast to earlier works, we derive a theoretical bound for individual learning intensity or t_{ls} within the PrMF. Section 2 outlines the formulation of the probabilistic models used in deriving the theoretical bound. Subsequently, we propose a novel **A**pproximate **Pr**obabilistic **M**emetic **F**ramework (APrMF) that adapts by governing the learning intensity of each individual according to a theoretical upper bound *while* the search progresses. Section 3 describes the APrMF in details. Section 4 presents the numerical study on search performance of APrMF while Section 5 compares the performance of APrMF with several recent state-of-the-art advanced evolutionary, memetic and hybrid algorithms. Finally, Section 6 concludes with some recommendations for further research.

2 A Probabilistic Memetic Framework for Non-Linear Optimization

Optimization is the study of the mathematical properties of optimization problems and the analysis of algorithms. It deals with the problem of minimizing or maximizing a mathematical model of an objective function such as cost, fuel consumption, etc., subject to a set of constraints. Here, we consider the general non-linear programming problem of the form:

- A target objective function $f(\mathbf{x})$ to be minimized or maximized
- A set of real variables, $\mathbf{x} \in \mathbb{R}^{ndim}$, $\mathbf{x}_{low} \le \mathbf{x} \le \mathbf{x}_{up}$, where $\mathbf{x}_{low}$ and $\mathbf{x}_{up}$ are the vectors of lower and upper bounds, respectively, and *ndim* representing problem dimensionality.
- A set of equality/inequality constraints $g_w(\mathbf{x})$ that allow the unknowns to take on certain values but exclude others. For example, the constraints may take the form of $g_w(\mathbf{x}) \le 0$, for $w = 1, 2, ..., b$, where b is the number of constraints.

2.1 Memetic Algorithm

At a macro-level, MA can be defined as a synergy of evolution and individual learning. In the case of a minimization problem, the aim of the memetic algorithm is to locate the *global minimum* $\mathbf{x}^*$ such that $\mathbf{x}^* = \arg\min_{\mathbf{x}} f(\mathbf{x})$, without violating the constraints imposed. The pseudo-code of a canonical Memetic Algorithm is outlined in Figure 1.

Procedure: Canonical Memetic Algorithm
Begin
  /* *Evolution - Stochastic Search Operators* */*
  Initialize: Generate an initial population;
  While (Stopping conditions are not satisfied)
    ➢ Evaluate all individuals in the population.
    ➢ Select the subset of individuals, Ω_{il}, that should undergo the individual learning procedure.
      For each individual in Ω_{il}
        /* *Individual Learning – Local heuristics or Conventional exact methods* */*
        • Perform individual learning using meme(s) with probability of P_{il} or frequency f_{il} for a t_{il} period.
        • Proceed with Lamarckian or Baldwinian learning.
      End For
    ➢ Generate a new population using stochastic search operators.
  End while
End

Figure 1: Outline of a Memetic Algorithm

Without loss of generality, evolution usually involves some stochastic operators while individual learning is in the form of local heuristics [34] or conventional exact enumerative methods [35]-[38]. Examples of the stochastic evolutionary search include Monte Carlo Algorithm, Simulated Annealing, Genetic Algorithm, Swarm Algorithm and others. In practice, the objective function of a problem does not display convexity, thus several local optima in the form of minima and/or maxima may exist. In this case, a *local minimum* $\mathbf{x}_l$ is defined as a point for which there exists some $\delta > 0$ so that for all $\mathbf{x}$ such that $\left\| \mathbf{x} - \mathbf{x}_l \right\| \le \delta$, the expression $f(\mathbf{x}_l) \le f(\mathbf{x})$ holds. Examples of individual learning strategies include the hill climbing, Simplex method, Newton/Quasi-Newton method, interior point methods, conjugate gradient method, line search and other local heuristics.

In what follows, we present the theoretical formulations of probabilistic models for evolution and individual learning in MA. In particular, we model MA as a process involving the decision of embracing the separate actions of *evolution* or *individual learning* and analyze the probability of each process in locating the global optimum. To begin, we outline some basic definitions used in our formulations.

Definition 1: The optimization problem is considered solved or reached the global optimum if at least one solution $\mathbf{x}'$ satisfies the condition:

$$f(\mathbf{x}') \leq f(\mathbf{x}^*) + \varepsilon \qquad (1)$$

where ε denotes a very small value.

Definition 2: A type I point satisfies inequality (1). As shown in Figure 2, a point is of type I if it lies in the segment AB of the graph.

Definition 3: A type II point lies in a basin of attraction containing type I points. Based on Figure 2, type II points lie in the segment CD.

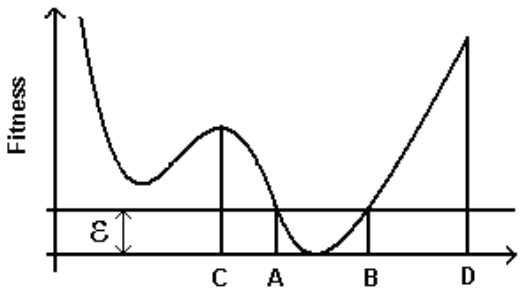

Figure 2: Illustrations of type I and type II points

Definition 4: $p_1^{(k)}$ or $p_2^{(k)}$ is the probability that an individual in the population of candidate solutions at generation *k* is a type I or type II point, respectively.

Note that it follows from the above definitions that all individual learning approaches, when started from a type II point always converges to a type I point within tractable computational budget. In Figure 2 the basins are shown as valley, but without loss of generality, the basin of attraction can be in the form of any neighborhood/modality structure.

Here, we model *evolution* and *individual learning* in MA as independent processes and analyze the probability of each process in locating type I point(s). Consider current generation *k*, the probability of finding at least one type I as a result of individual learning can be derived as:

$$P_l = 1 - (1 - p_2^{(k)})^{f_{il} * n} \qquad (2)$$

The total computational cost, C_{il} (in terms of function evaluation counts/calls), of individual learning in each search generation can be written as

$$C_{il} = t_{il} \times (f_{il} \times n) \tag{3}$$

where n represents the MA population size, t_{il} is the computational budget allocated to a single iteration of individual learning, and f_{il} is the frequency of individual learning that defines the portion of a population that undergoes individual learning.

For the same computational budget expended, the number of search generations Δ_g that one may replace individual learning with stochastic evolutionary search is given by:

$$\Delta_g = \frac{t_{il} \times (f_{il} \times n)}{t_{gs}} \tag{4}$$

where t_{gs} is the computational cost incurred by evolution in generating the subsequent populations. For the sake of consistency, we consider t_{gs} in terms of function evaluations incurred per generation. By assuming weak dependency among the value of $p_1^{(k)}$ across Δ_g generations, the probability P_g of having at least one type I point is given by:

$$P_g = 1 - \prod_{i=1}^{\Delta_g} \left(1 - p_1^{(k+i)}\right)^n \tag{5}$$

2.2 A Theoretical Upper Bound for t_{il} in MA

Here, we generalize the design of an MA as a decision of *evolution* against *individual learning* at each point in time. Clearly, individual learning should be used if it has a higher probability of reaching a type I point over stochastic evolution under equal computational budget, i.e.,

$$\begin{aligned} & P_l \geq P_g \\ \Leftrightarrow\ & 1 - \left(1 - p_2^{(k)}\right)^{f_{il} * n} \geq 1 - \prod_{i=1}^{\Delta_g} \left(1 - p_1^{(k+i)}\right)^n \\ \Leftrightarrow\ & \left(1 - p_2^{(k)}\right)^{f_{il} * n} \leq \prod_{i=1}^{\Delta_g} \left(1 - p_1^{(k+i)}\right)^n \end{aligned} \tag{6}$$

Every search algorithm, except for uniform random search, introduces some unique form of bias, suitable for some classes of problems but not for others. Hence it is fair to assume that the global search method chosen for solving the problem of interest is one that is deemed as capable of directing the search towards the promising region containing Type I points as the search evolves, i.e.,

$$\prod_{i=1}^{\Delta_g}\left(1-p_1^{(k+i)}\right)^n \le (1-p_1^{(k)})^{n*\Delta_g} \tag{7}$$

From (6) and (7) we have

$$\begin{aligned}&\left(1-p_2^{(k)}\right)^{f_{il}*n} \le \prod_{i=1}^{\Delta_g}\left(1-p_1^{(k+i)}\right)^n \le \left(1-p_1^{(k)}\right)^{n*\Delta_g}\\ &\Leftrightarrow \left(1-p_2^{(k)}\right)^{f_{il}} \le \left(1-p_1^{(k)}\right)^{\Delta_g}\end{aligned} \tag{8}$$

Taking logarithms of both sides, we arrive at:

$$\begin{aligned}&f_{il}\ln\left(1-p_2^{(k)}\right) \le \Delta_g \ln\left(1-p_1^{(k)}\right)\\ &\Leftrightarrow f_{il}\ln\left(1-p_2^{(k)}\right) \le \frac{t_{il}f_{il}n}{t_{gs}}\ln\left(1-p_1^{(k)}\right)\\ &\Leftrightarrow \quad \ln\left(1-p_2^{(k)}\right) \le \frac{t_{il}n}{t_{gs}}\ln\left(1-p_1^{(k)}\right)\end{aligned} \tag{9}$$

Since $\ln\left(1-p_1\right) < 0$, the above expression becomes:

$$t_{il} \le \frac{t_{gs}}{n}\frac{\ln\left(1-p_2^{(k)}\right)}{\ln\left(1-p_1^{(k)}\right)}$$

$$\text{Or } t_{il}^{upper} = \frac{t_{gs}}{n}\frac{\ln\left(1-p_2^{(k)}\right)}{\ln\left(1-p_1^{(k)}\right)} \tag{10}$$

Equation (10) thus provides a theoretical upper bound on the computational cost allowable for performing individual learning in MA. This implies that one should not allocate a computational budget greater than $\frac{t_{gs}}{n}\frac{\ln\left(1-p_2^{(k)}\right)}{\ln\left(1-p_1^{(k)}\right)}$ for conducting individual learning at generation *k*. Indirectly, it also implies that if a meme or learning procedure requires a computational budget of more than t_{il}^{upper} to reach a type I point, then it should not be used since it does not offer any competitive advantage over the process of evolution in the MA search. It is also worth noting that equation (10) is independent of f_{il} indicating that the frequency of learning in MA does not affect the theoretical bound for t_{il}.

3 Probabilistic Memetic Framework

In this section, we formulate a **Pr**obabilistic **M**emetic **F**ramework (PrMF) as a basis for adaptation. This is achieved by governing the maximum intensity of the individual learning using the theoretical bound derived in Section 2, *while* the search progresses. Based on the taxonomy in [22], PrMF represents an online adaptive approach. The pseudo code for PrMF is outlined in Figure 3. In the first step, the PrMF population is initialized either randomly or using design of experiments techniques such as Latin hypercube

sampling [39]. The evaluated populations of chromosomes then undergo evolution based on the stochastic operators. For instance, in a genetic based MA, the stochastic operators include mutation, crossover and selection.

Subsequently, in contrast to the canonical MA, the theoretical upper bound for learning intensity, t_{il}^{upper}, is estimated for each individual/chromosome in the newly evolved population, based on $p_1^{(k)}$, $p_2^{(k)}$ and $\frac{t_{gs}}{n}$ in equation (10). Each chromosome then undergoes individual learning using the specified meme according to their respective maximum allowable intensity. In the next subsection, we analyze the efficacy of the proposed PrMF using two synthetic benchmark functions with known fitness landscape.

Procedure: PrMF
Begin
/* *Evolution - Stochastic Search Operators* */
Initialize: Generate an initial population;
While (Stopping conditions are not satisfied)
- Evaluate all individuals in the population.

For each individual in new population

/* *Individual learning with t_{il} defined by the estimated theoretical upper bound* */

- Estimate the theoretical individual learning intensity bound, $t_{il}^{upper} = \frac{t_{gs}}{n} \frac{\ln\left(1-p_2^{(k)}\right)}{\ln\left(1-p_1^{(k)}\right)}$
- Perform individual learning using the specified meme for t_{il}^{upper} evaluations
- Proceed with Lamarckian or Baldwinian learning

End For
- Generate a new population using stochastic search operators.

End while
End

Figure 3: Outline of Probabilistic Memetic Framework.

3.1 *Unimodal* Sphere function

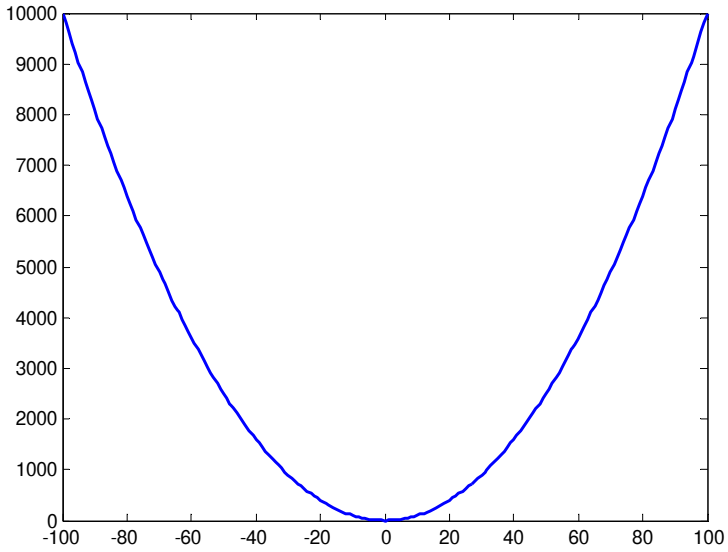

Figure 4: Unimodal Sphere function.

We first consider the Sphere function with a one-dimensional landscape depicted in Figure 4. For unimodal Sphere function, all decision vectors in the entire solution space including the global optimum lies in the single basin of attraction for a first or second-order derivative based local search optimization method. Hence, the probabilities of hitting type I ($p_1^{(k)}$) or type II ($p_2^{(k)}$) points at generation k can be easily estimated as $p_1^{(k)} < 1$ and $p_2^{(k)} = 1$, respectively. Note that $p_1^{(k)}$ approaches zero for increasing dimensionality of the Sphere function. With $p_1^{(k)} \sim 0$ and $p_2^{(k)} = 1$ in equation (10), the theoretical upper bound of t_{il} approaches infinity. This implies that all individual learning involving first or second-order derivative based methods would search more efficiently than random sampling or stochastic search methods in reaching the global optimum. From equation (2), it can also be derived that the probability P_l of finding at least one type I point or the global optimum of the Sphere function in the case of a unimodal problem when using individual learning, is 1, since $P_l = 1 - \left(1 - p_2^{(k)}\right)^{f_{il} * n} = 1$, as $p_2^{(k)} = 1$. Statistically, it makes better sense to use a first or second-order derivative based method over stochastic evolutionary operators to search on the unimodal Sphere function.

3.2 *Multi-Modal* Step function

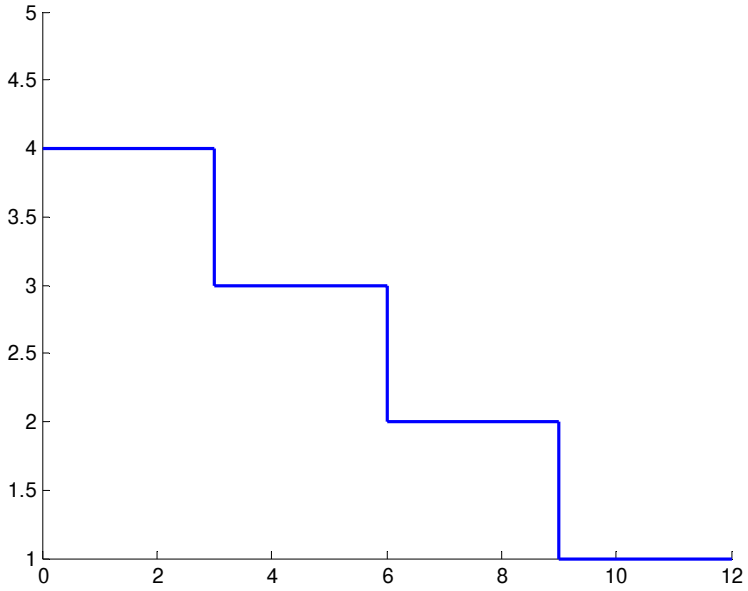

Figure 5: Multi-Modal Step function.

Next we consider the multi-modal Step function where the one-dimensional landscape is depicted in Figure 5. Consider again a first or second-order derivative based local search optimization method as the individual learning procedure versus the stochastic evolutionary operators in the PrMF. From the landscape in Figure 5, it can be easily observed that the set of type II points, for instance in the range [9, 12], is exactly the same as the set of type I points. As a result, for a specific search method considered, the probability of hitting type I ($p_1^{(k)}$) or type II ($p_2^{(k)}$) is equal, i.e., $p_1^{(k)} = p_2^{(k)}$, for all *k*. With $p_1^{(k)} = p_2^{(k)}$, the theoretical upper bound t_{il}^{upper} becomes $\frac{t_{gs}}{n}$ according to equation (10), i.e., $t_{il}^{upper} = \frac{t_{gs}}{n}\frac{\ln(1-p_2^{(k)})}{\ln(1-p_1^{(k)})} = \frac{t_{gs}}{n}$. Further, the condition $t_{gs} \le n$ holds since the entire or a portion of the population is generally replaced and evaluated in each search generation. With $t_{gs} \le n$ and t_{il}^{upper} becomes ≤ 1 indicating that derivative-based individual learning should not be performed on this multi-modal Step problem. Note that first or second-order derivative based methods do not generate improvements on a stationary point and since all points in the search space and basin of attraction are stationary, i.e., (gradient is equal to 0), any amount of computational budget allocated to a derivative-based individual learning is unlikely to contribute to the success of PrMF in search for the global optimum.

3.3 Approximate Probabilistic Memetic Framework

In the previous section, we have assumed that information on $p_1^{(k)}$ and $p_2^{(k)}$ for the problem in hand can be easily obtained or estimated. In reality, one usually has little or no prior knowledge of $p_1^{(k)}$ and $p_2^{(k)}$ in complex optimization problems. In this regard, we propose a novel **A**pproximate **Pr**obabilistic **M**emetic Framework (APrMF) that incorporates an *Individual Learning Intensity Estimation Scheme* for approximating $p_1^{(k)}$, $p_2^{(k)}$ and the theoretical upper bound of the problem in hand. In doing so, APrMF adapts/governs the individual learning intensity of each individual using the *approximated* theoretical upper bound as the search progresses.

To begin, Figure 6 presents an outline of the APrMF. In the first step, the APrMF population is initialized either randomly or using design of experiments techniques such as Latin hypercube sampling. During this initial stage, the APrMF search operates similarly to the canonical MA with all chromosomes assigned equal computational budget for conducting individual learning. Nevertheless, in contrast to a canonical MA, the individual learning procedure or meme in APrMF is equipped with search tracking capability such that the search history and structure of each chromosome is archived in database Φ. The archived search history is then used for estimating the theoretical upper bound of future individual learning intensity.

Procedure: APrMF
Begin

/* *Start of Canonical MA* */
Initialize: Generate an initial population;
For the first few generations
- Evaluate all individuals in the population.

For each individual $\mathbf{x}(i)$ in current population
- Perform individual learning using the specified *meme with tracking capabilities* for a maximum of $t_{il}(i) = t_{il}^{initial}$ evaluations
- Proceed with Lamarckian or Baldwinian learning

End For
- Generate a new population using stochastic search operators.

End For

/* *End of Canonical MA* */

While (Stopping conditions are not satisfied)
- Evaluate all individuals in the population

For each individual $\mathbf{x}(i)$ in new population

/* *Individual learning with adaptive* t_{il} *set according to upper bound and the expected value* */
- Estimate learning intensity upper bound $t_{il}^{upper}(i)$ and expected learning intensity $t_{il}^{expected}(i)$ for individual $\mathbf{x}(i)$ using the *Individual Learning Intensity Estimation Scheme* outlined in Figure 7.
- **If** ($t_{il}^{expected}(i) \le t_{il}^{upper}(i)$) **then**

/* *increase budget for individual learning* */

- $t_{il}(i) = f(t_{il}^{expected}(i))$
- Perform individual learning using the specified meme for $t_{il}(i)$ evaluations
- Proceed with Lamarckian or Baldwinian learning

Else

/* *Do not perform any individual learning* */

End If

End For

➢ Generate a new population using stochastic search operators.

End while

End

Figure 6: Outline of the Approximate Probabilistic Memetic Framework.

After some pre-defined number of generations has elapsed, the adaptive mechanism for *individual learning intensity*, i.e., t_{il}, takes effect. For each chromosome/individual, denoted here as $\mathbf{x}(i)$, that undergoes individual learning, the upper learning intensity bound $t_{il}^{upper}(i)$ and expected learning intensity $t_{il}^{expected}(i)$ are estimated based on the *Individual Learning Intensity Estimation Scheme* outlined in Figure 7. Here, we describe the *Individual Learning Intensity Estimation Scheme* as six main steps with the aid of the simple illustration in Figure 8:

- Step 1: Locate the q nearest neighbors of individual $\mathbf{x}(i)$ from database Φ based on some appropriate distance metric. For example, a simple Euclidean distance metric may be used in the case of a continuous parametric optimization problem, i.e., the two nearest neighboring points of individual $\mathbf{x}$ in Figure 8(a), are denoted by $\mathbf{x}_{\text{neighbor1}}$ and $\mathbf{x}_{\text{neighbor2}}$.

- Step 2: Identify the set Ω_{trace} of q learning search traces associated with the nearest q chromosomes. In Figure 8, the search traces for $\mathbf{x}_{\text{neighbor1}}$ and $\mathbf{x}_{\text{neighbor2}}$ are t_1 and t_2, respectively.

- Step 3: Determine the fittest point, $\mathbf{x}_{\mathbf{best}}$, in Ω_{trace}, i.e., $\mathbf{x}_{\mathbf{best}} = \arg\min_{\mathbf{x}} \{ f(\mathbf{x}) \mid \mathbf{x} \in \Omega_{trace} \}$. $\mathbf{x}_{\mathbf{best}}$ represents the best quality solution found so far within the local vicinity or neighborhood of $\mathbf{x}(i)$. $\mathbf{x}_{\mathbf{best}}$ is denoted by point A in Figure 8.

- Step 4: Determine the range of type I point which is estimated based on the furthest $\varepsilon - close$ point in Ω_{trace} to $\mathbf{x}_{\mathbf{best}}$, denoted here as $\mathbf{x}_{\mathbf{1}}$, i.e., $\mathbf{x}_{\mathbf{1}} = \arg\max_{\mathbf{x}} \{ \| \mathbf{x} - \mathbf{x}_{\mathbf{best}} \| \mid f(\mathbf{x}) \le f(\mathbf{x}_{\mathbf{best}}) + \varepsilon \}$. In Figure 8, $\mathbf{x}_1$ is defined as point B.

- Step 5: The values of probabilities $p_1^{(k)}$ and $p_2^{(k)}$ for the neighborhood region of $\mathbf{x}(i)$ are approximated by:

$$p_2^{(k)} = \left\| \mathbf{x} - \mathbf{x_{best}} \right\|^{ndim} \Big/ volume\ of\ search\ space$$

$$p_1^{(k)} = \left\| \mathbf{x_1} - \mathbf{x_{best}} \right\|^{ndim} \Big/ volume\ of\ search\ space$$

The upper learning intensity bound t_{ls}^{upper} of the current individual is then estimated by:

$$t_{il}^{upper} = \frac{t_{gs}}{n} \frac{\ln(1 - p_2^{(k)})}{\ln(1 - p_1^{(k)})}$$

$$= \frac{t_{gs}}{n} \frac{\ln(1 - \left\| \mathbf{x} - \mathbf{x_{best}} \right\|^{ndim} / volume\ of\ search\ space)}{\ln(1 - \left\| \mathbf{x_1} - \mathbf{x_{best}} \right\|^{ndim} / volume\ of\ search\ space)}$$

Based on Taylor series expansion, $\ln(1 - \alpha^n) \approx -\alpha^n$ for small value of α, the above equation simplifies to:

$$t_{il}^{upper} = \frac{t_{gs}}{n} \frac{\left\| \mathbf{x} - \mathbf{x_{best}} \right\|^{ndim}}{\left\| \mathbf{x_1} - \mathbf{x_{best}} \right\|^{ndim}}$$

For the illustration example in Figure 8,

$t_{il}^{upper} = \frac{CA}{BA} = 5$ for the case I in Figure 8(a) while

$t_{il}^{upper} = \frac{CA}{BA} = 2$ for the case II in Figure 8(b).

- Step 6: Estimate the expected learning intensity $t_{il}^{expected}$ as an average of the intensity for the k nearest neighbors. In Figure 8, the expected intensity for individual $\mathbf{x}$ is 3 since its neighbors, defined by $\mathbf{x}_{\text{neighbor1}}$ and $\mathbf{x}_{\text{neighbor2}}$ took 3 function evaluations on average in their individual learning processes.

Procedure: Individual Learning Intensity Estimation Scheme
Begin

/ Estimate p_1 and p_2 */*

1. Identify set Ω, the q nearest chromosomes of $\mathbf{x}$ in database Φ.
2. Identify set Ω_{trace}, the q learning search traces associated with the q nearest chromosomes.
3. Find $\mathbf{x_{best}}$, the fittest individual in Ω_{trace}, i.e., $\mathbf{x_{best}} = \arg\min_{\mathbf{x}} \{ f(\mathbf{x}) \mid \mathbf{x} \in \Omega_{trace} \}$.
4. Find $\mathbf{x_1}$, the furthest ε-close point to $\mathbf{x_{best}}$, i.e., $\mathbf{x_1} = \arg\max_{\mathbf{x}} \{ \left\| \mathbf{x} - \mathbf{x_{best}} \right\| \mid f(\mathbf{x}) \le f(\mathbf{x_{best}}) + \varepsilon \}$.
5. Estimate the upper bound for learning intensity,

$$t_{ls}^{upper} = \frac{t_{gs}}{n}\frac{\ln\left(1-p_2^{(k)}\right)}{\ln\left(1-p_1^{(k)}\right)} = \frac{t_{gs}}{n}\frac{\left\|\mathbf{x}-\mathbf{x}_{\mathbf{best}}\right\|^{ndim}}{\left\|\mathbf{x}_1-\mathbf{x}_{\mathbf{best}}\right\|^{ndim}}$$

6. Estimate the expected value for learning intensity,

$t_{il}^{expected}$ = average length of the search traces in Ω_{trace}

End

Figure 7: Outline of Individual Learning Intensity Estimation Scheme.

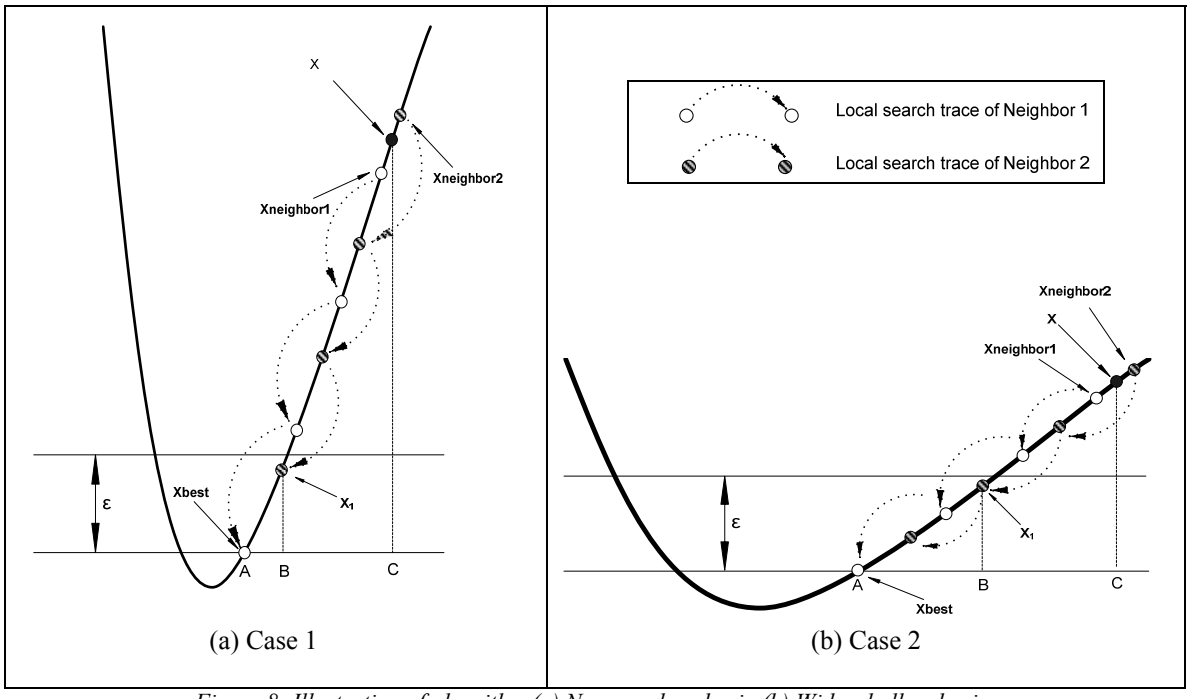

Figure 8: Illustration of algorithm (a) Narrow, deep basin (b) Wide, shallow basin.

For individual $\mathbf{x}(i)$, APrMF then proceeds with individual learning for a maximum computational budget defined by the estimated $t_{il}^{upper}(i)$ and $t_{il}^{\exp ected}(i)$, if the current expected learning intensity of the current neighborhood does not exceed the estimated learning intensity upper bound, i.e., $t_{il}^{expected}(i) \le t_{il}^{upper}(i)$. This is followed by a replacement of the genotype and fitness of $\mathbf{x}(i)$ in the population or only the latter action with the improved solution in the spirit of Lamarckian or Baldwinian learning, respectively. Otherwise, no form of individual learning would be carried out should $t_{il}^{expected} > t_{il}^{upper}$ happens. The stochastic GA operators are then used to create the next population. The entire process repeats until the specified stopping criteria are satisfied.

4 Empirical study

In this section, we present a numerical study to analyze the search behavior and performance of the

Approximate Probabilistic Memetic Framework. Several commonly used continuous parametric benchmark test problems already extensively discussed in the literature are used here to demonstrate the efficacy of the proposed probabilistic framework. They represent classes of unimodal/multimodal, discrete/continuous, epistatic/non-epistatic test functions. Table 1 tabulates these benchmark test functions with their notable characteristics.

To see how the proposed APrMF improves the efficiency of a search, we consider diverse candidates of MA formed by the synergy of different stochastic and individual learning strategies on the representative benchmark problems. In particular, we consider the i) Canonical Genetic Algorithm (GA) [3], or ii) Differential Evolution (DE) [40] or iii) Evolutionary Strategy (ES) [41] as candidate stochastic search strategies here. For individual learning or meme, we consider the i) strategy of Davies, Swann, and Campey with Gram-Schmidt orthogonalization (DSCG) [35] , ii) Davidon, Fletcher and Powell Strategy (DFP) [37] and iii) Simplex strategy of Nelder and Mead [36]. These are representative of second, first and zeroth order exact local search methods commonly found in the literature.

Benchmark Test Functions	**Range of x_i**	**Characteristics**			**Global Optimum**
		Epi*	**Mul***	**Disc***	
$F_{Sphere} = \sum_{i=1}^{D} x_i^2$	$[-100,100]^D$	none	none	none	0.0
$F_{Step} = 6D + \sum_{i=1}^{D} \lfloor x_i \rfloor$	$[-5.12, 5.12]^D$	none	none	medium	0.0
$F_{Griewank} = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-6, 6]^D$	weak	high	none	0.0
$F_{Ackley} = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2} - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20$	$[-32, 32]^D$	none	weak	none	0.0
$F_{Rastrigin} = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	$[-5, 5]^D$	none	high	none	0.0

Epi*: Epistasis Mul*: Multimodality Disc*: Discontinuity

Table 1: The multimodal benchmark functions used in the study (D is the number of dimensions).

To gain a better understanding of APrMF, we analyzed and compared with the canonical MA having different fixed individual learning intensity according to the following aspects:

- Search Quality and Efficiency - the capability of the strategy to provide high search quality and efficiency over different problem types.
- Computational Cost - the amount of extra CPU effort incurred over and above canonical memetic algorithm.

- Robustness - the capability of the framework/algorithm to generate performances that is reliable over different problems.
- Simplicity and ease of implementation - Simple framework/algorithm should require minimum effort to develop, as well as a minimum numbers of control parameters that need to be managed.

A. Search quality and efficiency

The average convergence trends obtained by APrMF based on GA and DFP individual learning procedure (GA-DFP) for solving the 30-dimension Sphere and Step problems are depicted in Figures 9 and 10, as a function of the total number of fitness evaluation calls. All results presented are averages over 25 independent runs. Each run continues until the global optimum was found or a maximum of 100,000 function evaluations was reached. In each run, the control parameters of the GA used in APrMF for solving the benchmark problems were set as follows: population size of 50, Gaussian mutation rate of 0.03, two-point crossover with a rate of 0.7 and real number encoding. The initial individual learning intensity used in APrMF is initialized to 100 evaluations. In Figures 9 and 10, canonical MA implies that throughout the entire search, a fixed individual learning intensity of 100 is used, i.e., no form of adaptation for t_{ls} is made. From the results reported in the figures, APrMF is shown to converge to the global optimum of both the Sphere and Step functions more efficiently than the canonical MA.

Besides the best fitness convergence trends obtained along the APrMF search, Figure 9(ii) also depicts the respective individual learning/global search ratio against fitness evaluation calls,. Note that the individual learning/global search ratio reflects the degree of local against the global efforts expended in the search so far. It is worth noting that during the initial stage of the plots in Figures 9 and 10 on learning/global search ratio against function evaluation call, both the APrMF and canonical MA exhibit similar trend or traces since the former operates exactly like the canonical MA during this initial learning phase of two search generations before its adaptive strategy begins to bite. From Figure 9(ii), it is observed that in contrast to the canonical MA (with a fixed individual learning intensity of 100) which reaches a stable state of individual learning/global search ratio, the individual learning/global search ratio of the APrMF increases significantly as the search progresses on the Sphere function. This implies that the expected individual learning intensity of the entire population increases adaptively once the knowledge about individual learning actually leads to greater benefits than the stochastic evolutionary method is gained. In contrast, the plot in Figure 10(ii) illustrates a case where the individual learning/global search ratio of the population drops since the converse is true in the case of a Step function, i.e., the use of the stochastic GA operators leads to greater benefits than individual learning. In both figures, APrMF is shown to converge to the global optimum more efficiently than canonical MAs. Note that these results obtained by APrMF based on GA and DFP is consistent with our theoretical analysis presented earlier in Section 3.3. Hence, this shows that the proposed Individual Learning Intensity Estimation Scheme in APrMF is capable of generating good approximation of p_1 , p_2 and t_{ls}^{upper} at search runtime.

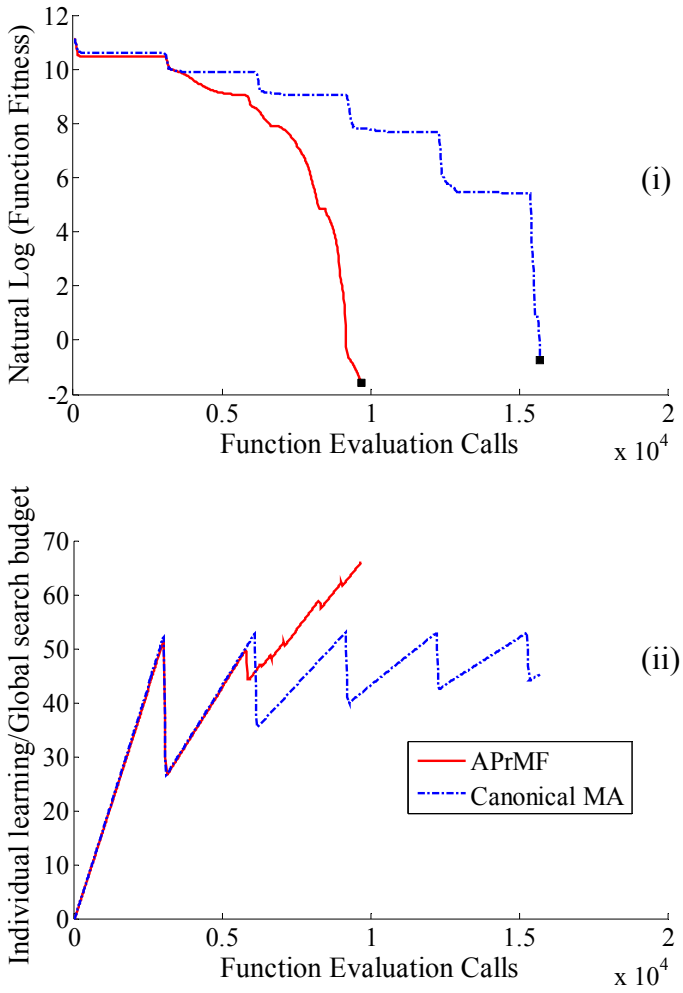

Figure 9: Search trends of Canonical MA (with a fixed individual learning intensity, $t_{il} = 100$) and APrMF (GA-DFP) on the Unimodal Sphere function.

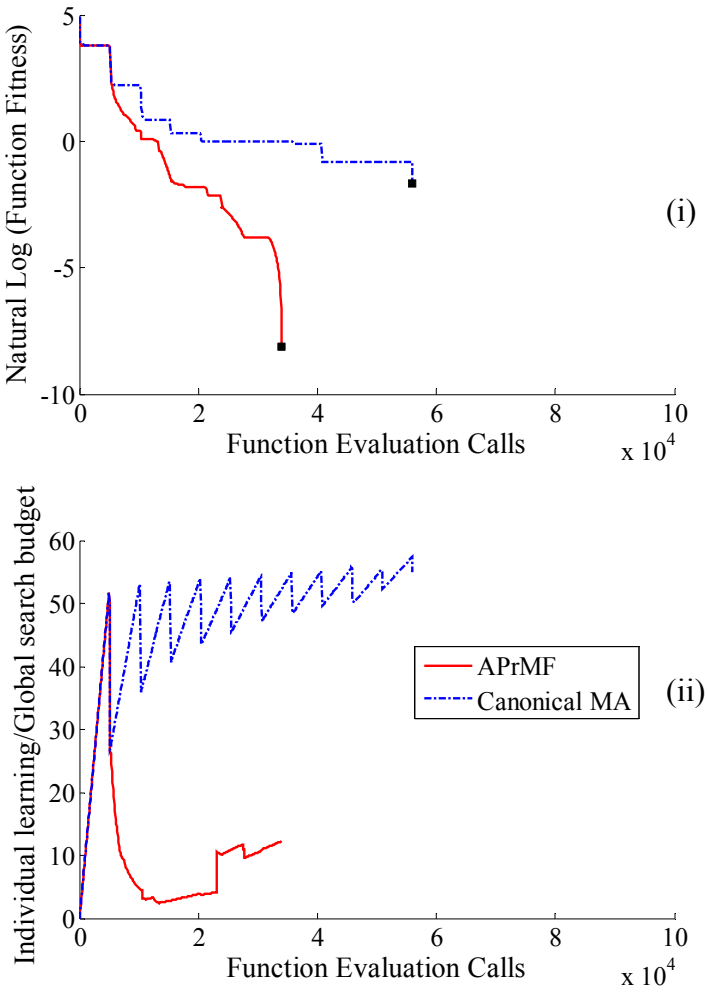

Figure 10: Search trends of Canonical MA (with a fixed individual learning intensity, $t_{il} = 100$) and APrMF (GA-DFP) on the Step function.

Next, we further analyze the search behavior of the APrMF for different evolutionary and individual learning procedures or memes on more benchmark problems that include the 30-dimensional Ackley, Griewank and Rastrigin functions. In particular, the following configurations of APrMFs and canonical MAs are investigated:

- APrMF with the *initial* individual learning intensity configured to 100 (APrMF1)
- APrMF with the *initial* individual learning intensity configured to 200 (APrMF2)
- Canonical Memetic Algorithm with *fixed* individual learning intensity of 100 (MA1)
- Canonical Memetic Algorithm with *fixed* individual learning intensity of 200 (MA2)

All parameters of the GA are kept consistent as before. Table 2 summarizes the parameter settings of the other search procedures used in the present experimental study.

General parameters	
Global search	GA, DE and ES
Local search	DSCG, DFP and Simplex
Stopping criteria	100,000 evaluations or convergence to global optimum
Population size	50
Genetic Algorithm parameters	
Encoding scheme	Real-coded
Selection scheme	Roulette wheel
Crossover operator	Two point crossover $p_c = 0.7$
Mutation operator	Gaussian mutation, $p_m = 0.03$
Differential Evolution parameters	
Crossover probability	$p_c = 0.9$
ES parameters	
Selection method	$\mu + \lambda$, $\mu = 50, \lambda = 100$
Mutation operator	Gaussian mutation
Local search parameters	
Initial local search intensity $t_{il}^{initial}$	100 or 200 evaluations

Table 2: Parameter setting of APrMF.

The averaged convergence trends of various MAs obtained for the 30-dimensional benchmark problems as a function of the total number of function evaluations are summarized in Figs. 11-13. For the sake of brevity in our discussion, the numerical results obtained are grouped accordingly in Figures 11, 12 and 13, so as to highlight the different adaptive trends of the individual learning/global search ratio and individual learning intensity in the APrM framework.

Note that all the subplots in Figure 11 share similar upward trends in the individual learning/global search ratio as the search progresses for different level of synergy between evolution and individual learning procedures on the benchmark problems. With the APrM framework, knowledge on the greater benefits of using individual learning against stochastic evolutionary search for the problems in hand is gained. Hence over time, the expected individual learning intensity or individual learning/global search ratio increases adaptively as the search progresses.

On the other hand, Figure 12 displays upward trends in the individual learning/global search ratio on APrMF1 while experiencing a downward trend on APrMF2. Since APrMF1 and APrMF2 differ in the configurations of initial individual learning intensity at 100 and 200, respectively, such trends serve to indicate that the optimum configuration of individual learning intensity should lie somewhere between the region from 100 to 200. Note that in Figure 12(b) and 12(d), the individual learning/global search ratios of APrMF1 and APrMF2 converged to very similar values. This serves to demonstrate that the proposed APrM framework is capable of adapting the individual learning intensity to suit the search problem in hand, regardless of its initial settings as well as the evolutionary and individual learning procedures used.

Figure 13 illustrates the inability of the individual learning procedures in contributing to the optimization problems. For instance, the Simplex individual learning procedure is found to synergize poorly with ES or DE in contributing to the MA search on the Griewank function, see Figure 13(a) and (c). Similarly, ES also did not synergize well with Simplex to form an MA that is appropriate for solving the Ackley function, as shown in Figure 13(b). Consequently, the APrM frameworks, i.e., APrMF1 and APrMF2, adapt by reducing the expected individual learning intensity of the population as the search proceeds, when knowledge on the inability of individual learning compared to the stochastic GA operators is learned. Thus a downward trend in the individual learning/global search ratio can be observed in Figures 13(a)-(c).

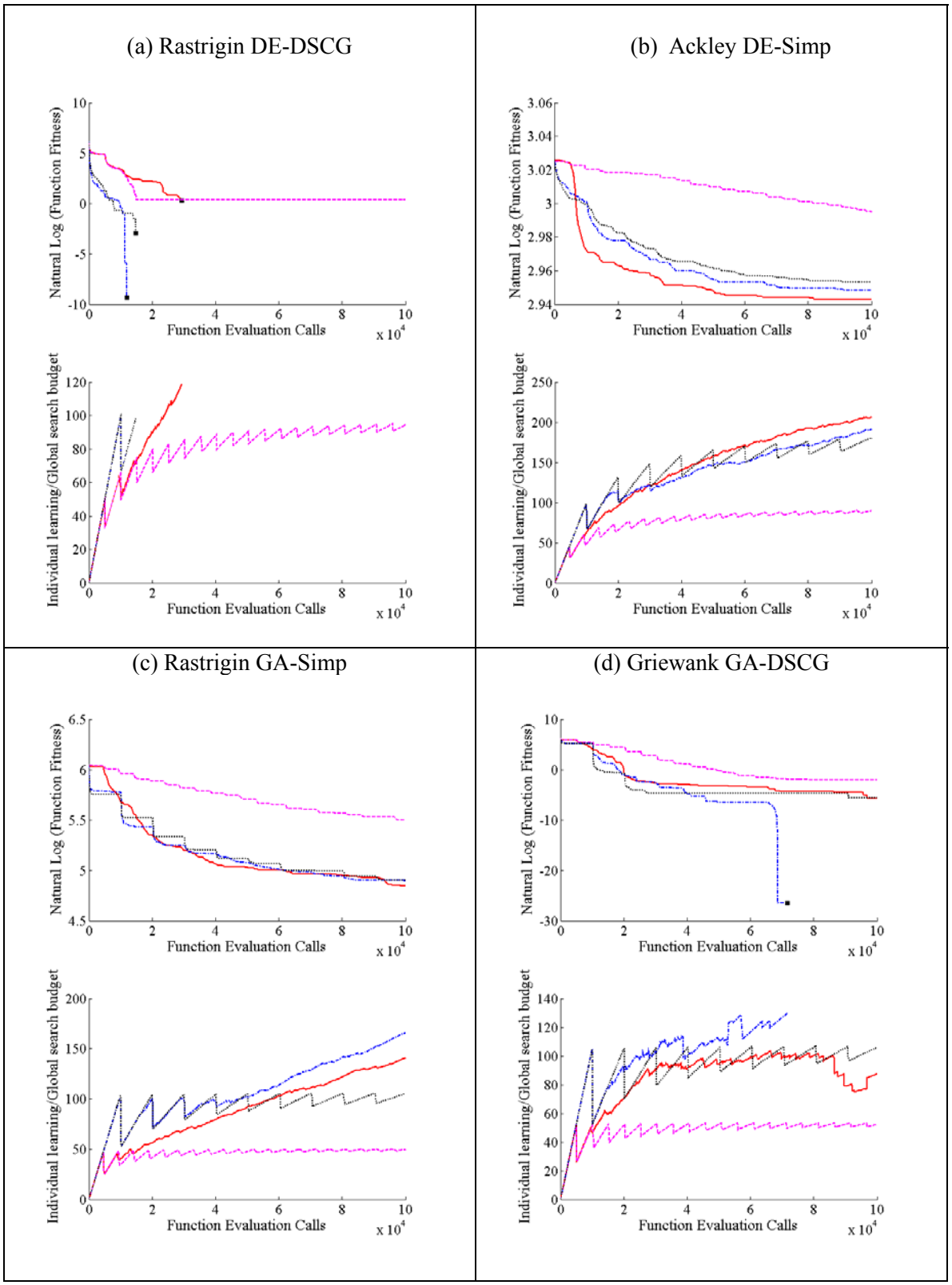

Figure 11: APrMF with Upward Trends in Individual learning/global search Ratio and t_{il} (for legends, refer to figure 13(d))

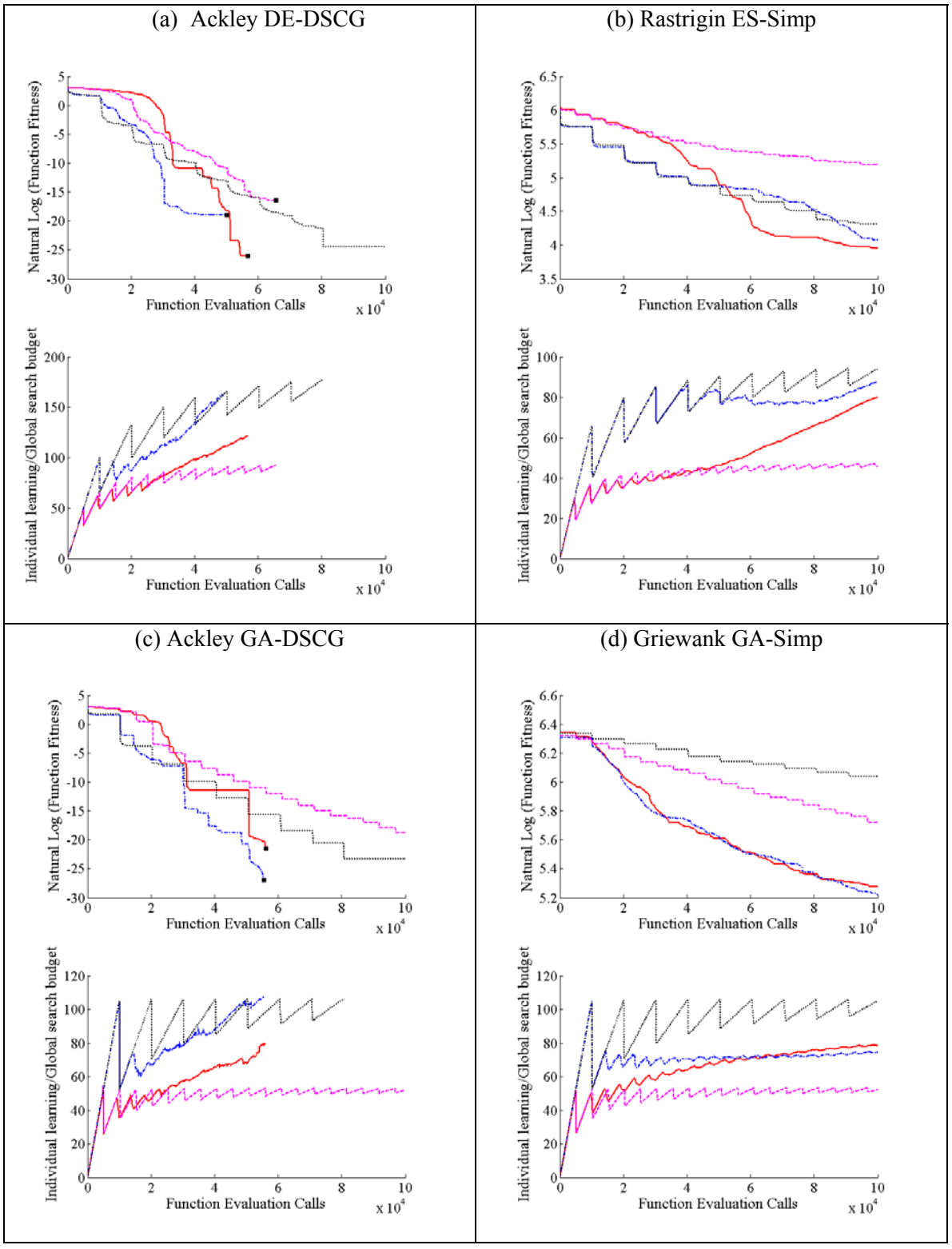

Figure 12: APrMF Adapting the Individual learning/global search Ratio to its Optimal Configuration t_{il} (for legends, refer to figure 13(d))

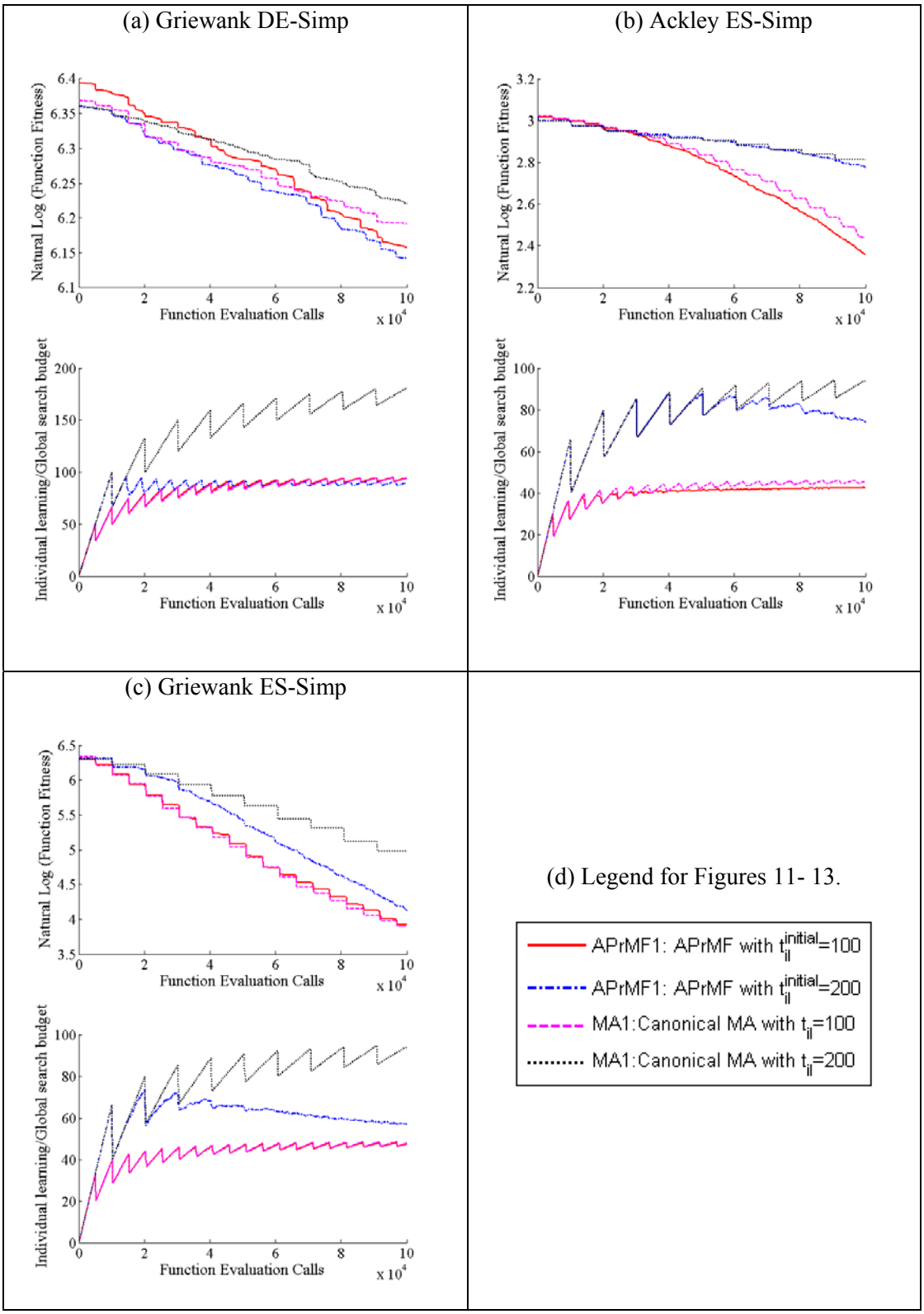

Figure 13: APrMF with Downward Trends in Individual learning/global search Ratio and t_{il} .

B. Computational complexity

The time complexity of a canonical Memetic Algorithm can be easily derived as the order of $O\{[n_{gen} \times t_{gs} \times \tau_{eval}] + [(n_{gen} \times n) \times t_{il} \times \tau_{eval}]\}$, where n_{gen} is the maximum number of generations evolved during the search process, τ_{eval} is the time taken for one objective function evaluation while t_{gs}, n and t_{il} have been defined in the previous sections. Hence, $n_{gen} \times popsize$ represents the total number of chromosomes that undergo individual learning in the entire search.

APrMF enhanced the canonical MA by introducing the *Individual Learning Intensity Estimation Scheme* which is made up of two main parts:

i) Identify the k nearest chromosomes, for which the time complexity is of the order of $O\{(n_{gen} \times n)^2 \times \tau_{dis}\}$ where τ_{dis} is the time taken to perform the distance measure between any two chromosomes.

ii) γ is the time to estimate the upper bound and expected learning intensity. The total time complexity incurred throughout the APrMF search is of the order of $O\{(n_{gen} \times n) \times \gamma\}$.

Hence, the total time complexity incurred by APrMF over the canonical MA is of the order of $O\{[(n_{gen} \times n)^2 \times \tau_{dis}] + [(n_{gen} \times n) \times \gamma]\}$. Note that since τ_{dis} and γ are relatively small (order of μsec) especially on complex optimization problems where function evaluations are computationally expensive [10], i.e., τ_{eval} is usually large (order of minutes or more), the extra cost incurred may be considered to be negligible.

C. Simplicity & Robustness

In contrast to the f_{il}, Ω_{il}, and t_{il} control parameters in a Canonical MA, APrMF represents a much simpler framework since it has only a single $t_{il}^{initial}$ parameter. Furthermore, from the results reported in Figures 11-13, the APrMF has been demonstrated to adapt the local search intensity t_{il} competently at runtime, regardless of the initial setting for $t_{il}^{initial}$, thus producing search performances that are superior to the canonical MA counterparts on all the benchmark problems considered. For instance, for the same initial configuration, APrMF is shown to increase the value of t_{il} to suit the Sphere landscape, while the reverse is observed on the Step function. In another example, both the APrMF1 and APrMF2 converged to robust search performances and global/individual learning ratio on the Griewank function, see Figure 12d. Hence, APrMF not only converges to robust solutions, but also offers ease of implementations as an added advantage.

5 APrMF with Comparison to Other Evolutionary and Memetic Approaches

In this section we provide a comprehensive empirical study of APrMF and a detailed comparison to several recent state-of-the-art evolutionary, memetic and/or hybrid evolutionary local search approaches using the test suite proposed in the Congress on Evolutionary Computation (CEC'05) [42]. These are comprehensive collections of unimodal/multimodal, discrete/continuous, epistatic/non-epistatic and hybridized test functions used for comparison in literature. The list of functions and their characteristics are summarized here in Table 3.†

The search performance of APrMF based on GA-DSCG for the 10D and 30D versions of the benchmark functions are summarized in Tables 5 and 6, respectively. In the tables, results are presented for 10^3, 10^4, 10^5 and $D*10^4$ function evaluations (D is the dimensionality of the benchmark problems considered). At each of the evaluation points, the objective function errors of 25 independent runs are sorted and the best, the 7th, the median, the 19th and the worst results together with the mean and standard deviation are presented in the tables. The parametric configuration of the GA-DSCG used is also summarized in Table 4.

Table 3: Benchmark functions for Real Number Optimization

Func	Benchmark Test Functions	Range	Characteristics		
			Epi*	Mul*	Disc*
1	$F_{Sphere} = \sum_{i=1}^{D} z_i^2$	$[-100,100]^D$	none	none	none
2	$F_{Schwefel1.2} = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} z_j \right)^2$	$[-100,100]^D$	high	none	none
3	$F_{Elliptic} = \sum_{i=1}^{D} \left(10^6\right)^{\frac{i-1}{D-1}} z_i^2$	$[-100,100]^D$	none	none	none
4	$F_{Schwefel1.2+Noise} = \left(\sum_{i=1}^{D} \left(\sum_{j=1}^{i} z_j \right)^2 \right) * \left(1 + 0.4\left\lvert N(0,1) \right\rvert\right)$	$[-100,100]^D$	high	high	none
5	$F_{Schwefel2.6} = \max\left\{ \left\lvert A_i x - A_i o \right\rvert \right\} \quad i = 1..D$	$[-100,100]^D$	none	none	medium
6	$F_{Rosenbrock} = \sum_{i=1}^{D-1} \left(100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2\right)$	$[-100, 100]^D$	high	high	none
7	$F_{Rastrigin} = \sum_{i=1}^{D} \left(z_i^2 - 10\cos(2\pi z_i) + 10\right)$	$[-5.12, 5.12]^D$	none	high	none
8	$F_{Griewank-R} = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1$	$[-\infty, +\infty]^D$	weak	high	none

† Due to space constraints, we present only the results of GA-DSCG on the 10D and 30D benchmark functions in this paper. For the results on other modes of synergy between evolutionary and individual learning MA, the reader may contact the authors for details.

9	$F_{Ackley-R} = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) + 20$	$[-32, 32]^D$	high	weak	none
10	$F_{Rastrigin-R} = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10)$	$[-5.12, 5.12]^D$	high	high	none
11	$F_{Weierstrass-R} = \sum_{i=1}^{D}(\sum_{k=0}^{k\max}[a^k \cos(2\pi b^k (z_i + 0.5))]) - D\sum_{k=0}^{k\max}[a^k \cos(\pi b^k)]$	$[-0.5, 0.5]^D$	high	high	none
12	$F_{Schwefel2.13} = \sum (A_i - B_i x)^2$, $A_i = \sum_{j=1}^{D}(a_{ij}\sin\alpha_j + b_{ij}\cos\alpha_j)$, $B_i = \sum_{j=1}^{D}(a_{ij}\sin x_j + b_{ij}\cos x_j)$	$[-\pi, \pi]^D$	weak	weak	none
13	$F_{GrieRos} = \sum_{i=1}^{D} F_{Griewank-R}(F_{Rosenbrock-R}(z_i, z_{i+1})), z_{D+1} = z_1$	$[-5, 5]^D$	high	high	none
14	$F_{Scaffer} = \sum_{i=1}^{D}(F(z_i, z_{i+1})), z_{D+1} = z_1$, $F(x,y) = 0.5 + \frac{\sin^2\left(\sqrt{x^2+y^2}\right) - 0.5}{\left(1 + 0.001\left(x^2+y^2\right)\right)^2}$	$[-100, 100]^D$	low	high	none
15	$F_{Hybrid1}$ (see f_{16} in [42])‡	$[-5, 5]^D$	high	high	medium
16	$F_{Hybrid2}$ (see f_{19} in [42])	$[-5, 5]^D$	high	high	medium

For rotated functions (8, 9, 10, 11 and 13): $\mathbf{z} = \mathbf{M}_{\mathbf{rotate}}(\mathbf{x} - \mathbf{o})$

For other functions: $\mathbf{z} = \mathbf{x} - \mathbf{o}$, where $\mathbf{o}$ is the shifted global optimum, $\mathbf{M}_{\mathbf{rotate}}$ is the rotation matrix

	10D functions	**30D functions**
Stopping criteria	100,000 evaluations	300,000 evaluations
Global search	Genetic algorithm	
Local search	DSCG	
Population size	50	
Encoding scheme	Real-coded	
Selection scheme	Roulette wheel selection	
Crossover operator	One point crossover $p_c = 0.7$	
Mutation operator	Gaussian mutation $p_m = 0.03$	
Initial local search intensity $t_{ls}^{initial}$	100 evaluations	300 evaluations

Table 4: Parameters configuration of APrMF based on GA-DSCG.

‡ Due to space constraints and the complexity of the definition, we do not present the equations of the hybrid functions here. Readers may refer to [42] for detailed explanations.

FEC		F_{Sphere}	$F_{Schwefel1.2}$	$F_{Elliptic}$	$F_{Schwefel1.2+Noise}$	$F_{Schwefel2.6}$	$F_{Rosenbrock}$	$F_{Rastrigin}$	$F_{Griewank-R}$
1e3	min	0	1.91e+3	2.59e-028	9.17e+3	2.09e+3	8.45e+0	0	7.63e-1
	7th	0	2.83e+3	1.42e-027	1.28e+4	5.74e+3	4.50e+1	0	9.57e-1
	med.	0	2.83e+3	5.25e-027	1.43e+4	6.70e+3	5.86e+1	0	9.57e-1
	19th	0	2.83e+3	1.13e-026	17420	7.71e+3	1.49e+2	0	9.57e-1
	max	0	3.50e+3	3.64e-026	2.28e+4	9.28e+3	1.39e+3	0	1.34e+0
	mean	0	2.80e+3	8.13e-027	1.50e+4	6.41e+3	2.82e+2	0	9.84e-1
	std	0	3.07e+2	8.33e-027	3.31e+3	1.80e+3	4.58e+2	0	1.11e-1
1e4	min	-	1.33e-6	-	2.27e+3	9.23e+1	0	-	0
	7th	-	6.34e-5	-	4.63e+3	7.35e+2	0	-	1.82e-10
	med.	-	6.34e-5	-	6.18e+3	1.66e+3	2.02e-7	-	2.46e-2
	19th	-	9.54e-5	-	7.15e+3	2.14e+3	1.03e+0	-	4.67e-2
	max	-	6.45e-3	-	1.02e+4	3.89e+3	4.94e+0	-	1.13e-1
	mean	-	4.01e-4	-	5.92e+3	1.54e+3	8.16e-1	-	2.95e-2
	std	-	1.28e-3	-	2.02e+3	9.26e+2	1.46e+0	-	2.96e-2
1e5	min	-	-	-	1.34e+2	7.36e-4	-	-	0
	7th	-	-	-	8.70e+2	6.48e+1	-	-	0
	med.	-	-	-	1.22e+3	1.53e+2	-	-	0
	19th	-	-	-	1.81e+3	2.57e+2	-	-	9.86e-3
	max	-	-	-	4.71e+3	1.20e+3	-	-	3.69e-2
	mean	-	-	-	1.48e+3	2.23e+2	-	-	8.07e-3
	std	-	-	-	1.01e+3	2.58e+2	-	-	1.14e-2

FEC		$F_{Ackley-R}$	$F_{Rastrigin-R}$	$F_{Weierstrass-R}$	$F_{Schwefel2.13}$	$F_{GrieRos}$	$F_{Scaffer}$	$F_{Hybrid1}$	$F_{Hybrid2}$
1e3	min	2.02e+1	6.78e+1	5.50e+0	1.35e+3	2.83e+0	1.23e-1	2.26e+2	9.10e+2
	7th	2.04e+1	7.98e+1	8.12e+0	2.13e+3	4.56e+0	3.97e-1	2.98e+2	1.03e+3
	med.	2.05e+1	7.98e+1	9.02e+0	3167	5.15e+0	4.42e-1	3.20e+2	1.07e+3
	19th	2.06e+1	7.98e+1	9.58e+0	3.78e+3	5.15e+0	6.16e-1	3.73e+2	1.11e+3
	max	2.08e+1	9.66e+1	1.08e+1	3.91e+3	6.25e+0	1.08e+0	4.71e+2	1.16e+3
	mean	2.05e+1	7.96e+1	8.67e+0	2.83e+3	4.81e+0	4.89e-1	3.37e+2	1.06e+3
	std	1.43e-1	5.43e+0	1.44e+0	8.52e+2	7.15e-1	2.21e-1	6.81e+1	6.72e+1
1e4	min	2.01e+1	2.79e+1	4.79e+0	1.01e-1	6.23e-1	3.89e-2	1.33e+2	6.76e+2
	7th	2.02e+1	3.88e+1	5.68e+0	2.80e+1	8.83e-1	5.83e-2	1.68e+2	8.42e+2
	med.	2.03e+1	4.49e+1	6.90e+0	5.73e+1	1.09e+0	7.79e-2	1.96e+2	9.09e+2
	19th	2.04e+1	5.55e+1	7.27e+0	9.25e+1	1.51e+0	1.e-1	2.11e+2	9.59e+2
	max	2.05e+1	7.98e+1	8.39e+0	3.99e+2	3.39e+0	1.66e-1	2.73e+2	1.03e+3
	mean	2.03e+1	4.88e+1	6.63e+0	8.36e+1	1.27e+0	8.49e-2	1.95e+2	8.85e+2
	std	1.11e-1	1.34e+1	1.04e+0	8.82e+1	6.41e-1	3.05e-2	3.65e+1	9.50e+1
1e5	min	2.0e+1	9.95e+0	4.17e+0	0	1.92e-1	1.94e-2	1.16e+2	4.32e+2
	7th	2.02e+1	1.99e+1	5.17e+0	0	5.84e-1	3.89e-2	1.31e+2	6.64e+2
	med.	2.02e+1	2.29e+1	5.60e+0	0	6.91e-1	3.89e-2	1.37e+2	7.90e+2
	19th	2.03e+1	2.80e+1	6.32e+0	1.84e-4	8.27e-1	7.77e-2	1.44e+2	8.e+2
	max	2.04e+1	3.38e+1	7.92e+0	3.52e-1	1.16e+0	9.72e-2	1.59e+2	8.26e+2
	mean	2.02e+1	2.30e+1	5.70e+0	2.38e-2	7.07e-1	4.99e-2	1.37e+2	7.21e+2
	std	1.05e-1	6.35e+0	1.01e+0	7.84e-2	2.07e-1	2.66e-2	1.18e+1	1.20e+2

Table 5: Optimization results using APrMF (GA-DSCG) on 10D benchmark functions. FEC represents Function Evaluation Calls incurred to converge at the global optimum solution of each respective function.

FEC		F_{Sphere}	$F_{Schwefel1.2}$	$F_{Elliptic}$	$F_{Schwefel1.2+Noise}$	$F_{Schwefel2.6}$	$F_{Rosenbrock}$	$F_{Rastrigin}$	$F_{Griewank-R}$
1e3	min	3.11e-27	3.85e+4	1.33e-24	7.67e+4	1.49e+4	2.60e+2	0	3.72e+0
	7th	1.31e-26	71670	1.30e-23	1569	1.94e+4	3.09e+3	4.01e+0	4.32e+0
	med.	3.26e-26	8.34e+4	2.71e-23	119853	2.17e+4	4.79e+3	5.31e+0	6.16e+0
	19th	3.26e-26	103222	8.15e-2	156792	2.45e+4	7.42e+3	2.89e+1	6.41e+0
	max	6.77e+2	107899	1.17e+1	173645	3.28e+4	1.15e+4	6.25e+1	6.41e+0
	mean	2.71e+1	8.30e+4	2.42e+0	1.27e+5	2.23e+4	5.29e+3	1.57e+1	5.51e+0
	std	1.35e+2	2.15e+4	4.55e+0	3.05e+4	4.32e+3	2.96e+3	1.80e+1	1.09e+0
1e4	min	-	7849	-	6.56e+4	1.06e+4	5.81e+1	0	4.22e-3
	7th	-	1.26e+4	-	8.52e+4	1.31e+4	2.25e+2	0	5.47e-1
	med.	-	1.41e+4	-	9.06e+4	1.43e+4	4.98e+2	0	6.21e-1
	19th	-	1.62e+4	-	107937	1.62e+4	1.14e+3	2.77e-2	7.23e-1
	max	-	2.70e+4	-	156792	2.41e+4	2.14e+3	5.98e+0	1.05e+0
	mean	-	1.52e+4	-	1.01e+5	1.47e+4	6.93e+2	6.66e-1	6.06e-1
	std	-	5.17e+3	-	2.54e+4	2.94e+3	5.84e+2	1.60e+0	2.41e-1
1e5	min	-	0	-	3.14e+4	5.93e+3	1.52e-7	-	0
	7th	-	0	-	41743	7.23e+3	8.82e+0	-	0
	med.	-	0	-	4.37e+4	8.37e+3	1.56e+1	-	0
	19th	-	0	-	4.90e+4	9.71e+3	2.86e+1	-	0
	max	-	8.41e+3	-	6.43e+4	1.13e+4	1.73e+2	-	9.86e-3
	mean	-	3.36e+2	-	4.55e+4	8.48e+3	3.30e+1	-	1.04e-3
	std	-	1.68e+3	-	6.91e+3	1.57e+3	4.10e+1	-	2.45e-3
3e5	min	-	-	-	2.57e+4	4.45e+3	0	-	0
	7th	-	-	-	3.15e+4	6.03e+3	1.34e-6	-	0
	med.	-	-	-	3.59e+4	6.39e+3	1.10e+0	-	0
	19th	-	-	-	4.03e+4	7.21e+3	1.32e+1	-	0
	max	-	-	-	5.42e+4	9.14e+3	5.79e+1	-	9.86e-3
	mean	-	-	-	3.61e+4	6.66e+3	8.27e+0	-	8.76e-4
	std	-	-	-	6.63e+3	1.28e+3	1.29e+1	-	2.36e-3

FEC		$F_{Ackley-R}$	$F_{Rastrigin-R}$	$F_{Weierstrass-R}$	$F_{Schwefel2.13}$	$F_{GrieRos}$	$F_{Scaffer}$	$F_{Hybrid1}$	$F_{Hybrid2}$
1e3	min	2.07e+1	3.83e+2	3.09e+1	4.22e+4	2.87e+1	8.51e-1	1.33e+2	8.47e+2
	7th	2.08e+1	4.41e+2	3.38e+1	7.07e+4	5.88e+1	2.82e+0	1.48e+2	9.07e+2
	med.	2.09e+1	4.80e+2	3.58e+1	120430	8.39e+4	3.14e+0	1.53e+2	9.31e+2
	19th	2.11e+1	4.86e+2	3.79e+1	140507	160182	3.93e+0	2.47e+2	9.57e+2
	max	2.11e+1	5.16e+2	3.86e+1	182762	162441	4.84e+0	5.21e+2	9.68e+2
	mean	2.09e+1	4.66e+2	3.58e+1	1.11e+5	7.58e+4	3.31e+0	2.16e+2	9.28e+2
	std	1.21e-1	3.71e+1	2.14e+0	4.e+4	7.14e+4	8.94e-1	1.19e+2	3.05e+1
1e4	min	2.05e+1	2.49e+2	2.10e+1	1.60e+4	7.98e+0	8.51e-1	1.16e+2	5.26e+2
	7th	2.07e+1	3.39e+2	3.04e+1	3.42e+4	1.37e+1	1.27e+0	1.23e+2	7.70e+2
	med.	2.08e+1	3.39e+2	3.12e+1	4.63e+4	2.06e+1	1.54e+0	1.27e+2	8.34e+2
	19th	2.08e+1	3.92e+2	3.23e+1	51674	3.16e+1	1.85e+0	1.31e+2	8.94e+2
	max	2.09e+1	4.69e+2	3.58e+1	7.44e+4	2.13e+2	2.67e+0	1.39e+2	9.45e+2
	mean	2.08e+1	3.64e+2	3.12e+1	4.44e+4	3.76e+1	1.59e+0	1.27e+2	8.17e+2
	std	9.90e-2	5.36e+1	2.79e+0	1.32e+4	5.30e+1	4.67e-1	6.21e+0	1.05e+2
1e5	min	2.e+1	1.79e+2	2.10e+1	3.29e+0	4.71e+0	2.45e-1	1.03e+2	5.03e+2
	7th	2.06e+1	2.34e+2	2.93e+1	4.41e+2	6.84e+0	6.29e-1	1.12e+2	5.75e+2
	med.	2.07e+1	2.51e+2	3.02e+1	1.63e+3	8.68e+0	7.72e-1	1.17e+2	5.98e+2
	19th	2.08e+1	2.82e+2	3.12e+1	2.79e+3	1.25e+1	9.29e-1	1.19e+2	6.56e+2
	max	2.09e+1	3.38e+2	3.32e+1	5.87e+3	2.22e+1	1.54e+0	1.21e+2	7.43e+2
	mean	2.07e+1	2.55e+2	2.98e+1	1.87e+3	9.89e+0	8.16e-1	1.15e+2	6.12e+2
	std	1.86e-1	4.29e+1	2.49e+0	1.70e+3	4.13e+0	3.26e-1	4.81e+0	5.87e+1
3e5	min	2.e+1	1.71e+2	2.10e+1	2.42e-6	4.71e+0	2.34e-1	1.03e+2	4.78e+2
	7th	2.06e+1	1.92e+2	2.82e+1	8.22e+0	6.28e+0	4.31e-1	1.07e+2	4.98e+2
	med.	2.07e+1	2.14e+2	2.99e+1	1.40e+2	6.84e+0	5.21e-1	1.13e+2	5.26e+2
	19th	2.08e+1	2.33e+2	3.04e+1	6.64e+2	8.16e+0	6.85e-1	1.16e+2	5.51e+2
	max	2.08e+1	2.91e+2	3.32e+1	2.83e+3	1.38e+1	8.76e-1	1.19e+2	6.45e+2
	mean	2.06e+1	2.17e+2	2.93e+1	4.42e+2	7.72e+0	5.41e-1	1.12e+2	5.34e+2
	std	1.91e-1	3.36e+1	2.54e+0	6.72e+2	2.26e+0	1.88e-1	5.12e+0	4.31e+1

Table 6: Optimization results using APrMF (GA-DSCG) on 30D benchmark functions. FEC represents Function Evaluation Calls incurred to converge at the global optimum solution of each respective function.

Using the results obtained in Tables 5 and 6, we pit APrMF against other memetic and hybrid evolutionary local search approaches. The list of algorithms used here for comparisons were reported in the recent CEC'05 which significantly outperform many others in the literature. Table 7 provides a brief description of these algorithms, while Tables 8 and 9 summarize their search performance compared to the APrMF in solving the same set of benchmark problems. For fair comparison, the same performance metric used in CEC'05 has been maintained. Entries in each table are the average number of function evaluations for all the successful runs (among 25 independent runs). For fair comparisons, the accuracy level of convergence, ε, is set to 10^{-6} for functions 1 to 5 and 10^{-2} for all others, similar to that used in the CEC 2005 benchmark studies. Overall, APrMF outperforms all the other methods on the unimodal, multimodal, discrete, continuous, epistatic, and non-epistatic 10 and 30 dimensional benchmark problems. On the hybridized problems, i.e., functions 15 and 16 in Table 3, competitive results have been obtained for APrMF compared to the other approaches considered.

Algorithm name	**Description**
BLX-GL50	Hybrid Real-Coded Genetic Algorithms [43]
BLX-MA	Real-coded memetic algorithm with adaptive local-search probability and local search length [44]
DMS-L-PSO	Dynamic multi-swarm particle swarm optimizer with local search [45]
EDA	Continuous Estimation of Distribution Algorithms [46]
DEshcSPX	Differential evolution with crossover-based local search [47]
G-CMAES	Restart CMA Evolution Strategy With Increasing Population Size [48]

Table 7: Memetic Algorithms or Hybrid EA-Local search used in Comparison.

	F_{Sphere}	$F_{\text{Schwefel 1.2}}$	F_{Elliptic}	$F_{\text{Rosenbrock}}$	$F_{\text{Griewank-R}}$	$F_{\text{Rastrigin}}$	$F_{\text{Schwefel2.13}}$
APrMF	***0.6(25)***	11.0(25)	***0.6(25)***	***10.5(25)***	***17.6(20)***	***1.0(25)***	***31.9(19)***
BLX-GL50	19.0(25)	41.04(25)	-	51.8(25)	20.8(9)	20.4(3)	51.59(13)
BLX-MA	12.0(25)	36.96(25)	-	-	-	69.8(18)	-
DMS-L-PSO	12.0(25)	12.0 (25)	11.7(25)	54.7(25)	94.8(4)	35.7(25)	54.1(19)
EDA	10.0(25)	11.0(25)	16.3(23)	68.2(25)	75.9(1)	-	35.2(10)
DEshcSPX	22.9(25)	34.7(25)	89.2(20)	50.2(23)	97.3(21)	89.7(5)	-
G-CMAES	1.61(25)	***2.38(25)***	6.5(25)	**10.8(25)**	-	4.67(25)	**32.7(19)**

*Table 8: Success measure of the algorithms in solving the 10D benchmark functions. For instance, 0.6 (25) on F_{Sphere} implies that APrMF incurred an average of (0.6*1000) function evaluation calls on 25 successful independent runs. A '–' entry in the table implies that the respective algorithm fails to converge to the global optimum. Bold italic also highlights the best search performance (based on pair-wise t-test between each respective algorithm pairs).*

	F_{Sphere}	$F_{\text{Schwefel 1.2}}$	F_{Elliptic}	$F_{\text{Rosenbrock}}$	$F_{\text{Griewank-R}}$	$F_{\text{Rastrigin}}$	$F_{\text{Schwefel2.13}}$
BLX-GL50	s+	s+	-	s+	s+	s+	s+
BLX-MA	s+	s+	-	-	-	s+	-
DMS-L-PSO	s+	s+	s+	s+	s+	s+	s+
EDA	s+	≈	s+	s+	s+	-	s+
DEshcSPX	s+	s+	s+	s+	s+	s+	-
G-CMAES	s+	s-	s+	≈	-	s+	≈

Table 9: Result of t-test with 95% confidence level comparing statistical values for APrMF and those of the other algorithms in solving the 10D benchmark functions (s+, s-, and ≈ indicate that APrMF is significantly better, significantly worse, and indifferent, respectively).

	F_{Sphere}	$F_{Schwefel\ 1.2}$	$F_{Elliptic}$	$F_{Griewank\text{-}R}$	$F_{Rastrigin}$
APrMF	***0.6(25)***	87.0(25)	***1.0(25)***	***25.5 (25)***	***5.9(25)***
BLX-GL50	58.05(25)	159.6(25)	-	66.3(25)	-
BLX-MA	32.13(25)	-	-	-	238.8(9)
DMS-L-PSO	5.13(25)	129.6(25)	285.3(21)	57.4(24)	-
EDA	150.1(25)	159.6(25)	219.3(25)	129.93(25)	-
DEshcSPX	89.4(25)	299.3(2)	-	148.1(21)	-
G-CMAES	4.5(25)	***13.0(25)***	42.7(25)	-	**6.1(25)**

Table 10: Success measure of the algorithms in solving the 30D benchmark functions. A '–' entry in the table implies that the respective algorithm fails to converge to the global optimum. Bold italic also highlights the best search performance (based on pair-wise t-test between each respective algorithm pairs).

	F_{Sphere}	$F_{Schwefel\ 1.2}$	$F_{Elliptic}$	$F_{Griewank\text{-}R}$	$F_{Rastrigin}$
BLX-GL50	s+	s+	-	s+	-
BLX-MA	s+		-	-	s+
DMS-L-PSO	s+	s+	s+	s+	-
EDA	s+	s+	s+	s+	-
DEshcSPX	s+	s+	-	s+	-
G-CMAES	s+	s-	s+	-	≈

Table 11: Result of t-test with 95% confidence level comparing statistical values for APrMF and those of the other algorithms in solving the 30D benchmark functions (s+, s-, and ≈ indicate that APrMF is significantly better, significantly worse, and indifferent, respectively).

A further comparative study of APrMF (also based on GA-DSCG for consistency) to recent advance evolutionary algorithms is also provided here and the results are summarized in Figure 14. In particular, we pit APrMF against the Fast Evolutionary Strategy[49], Fast Evolutionary Programming [50] Orthogonal GA (OGA/Q) [51], and Hybrid Taguchi-Genetic Algorithm (HTGA) [52] on common benchmark functions used that includes the non-rotated Ackley, Griewank and Rastrigin functions, described previously in Table 3. The results in Figure 14 indicate the superior performance of the APrMF in converging to the global optimum more efficiently than all the counterparts considered, three of which were published recently in the IEEE Transactions on Evolutionary Computation.

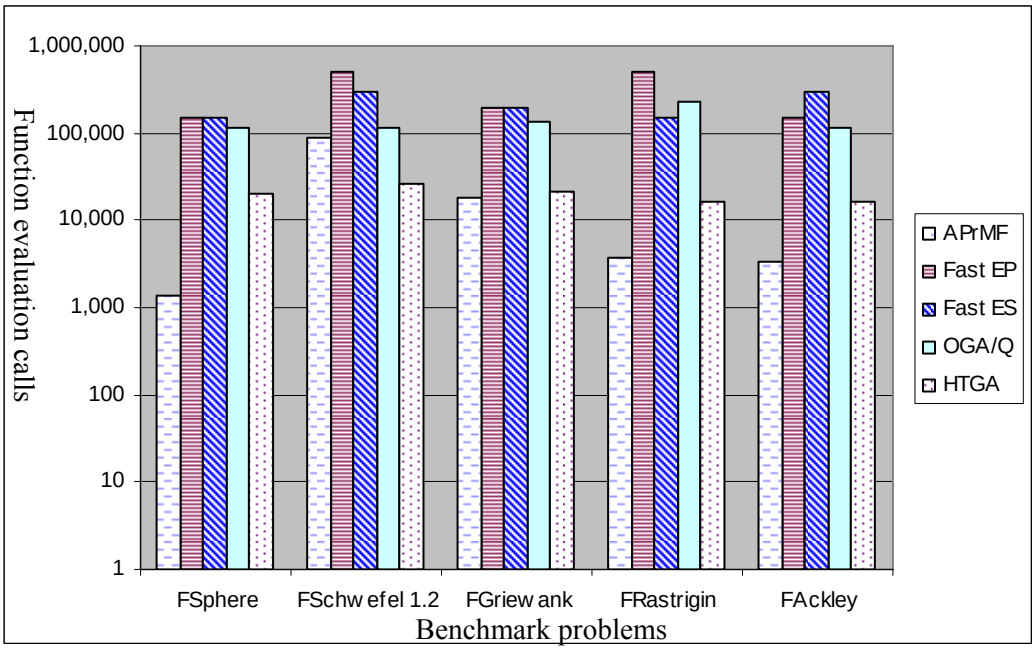

Figure 14: Number of function evaluations used by different algorithms in solving the 30D benchmark functions.

6 Conclusion

One major issue pertaining to adaptation is to determine during runtime whether evolution or individual learning should be directed, with the objective of accelerating MA search. In contrast to earlier works which rely on semi-adhoc or heuristics methods, the essential backbone of our framework is a probabilistic model derived for establishing the theoretical upper bound of individual learning intensity. In particular, we model MA as a process involving the decision of embracing the separate actions of evolution or individual learning and analyze the probability of each process in locating the global optimum. This leads to a novel **Pr**obabilistic **M**emetic **F**ramework, a quantitative formalization for adaptation by governing the learning intensity of each individual according to the theoretical upper bound while the search progresses.

An important aspect of PrMF is the determination of certain probabilistic measures based on the neighborhood structure of a local search. We showed in this paper that in practice, such probabilistic measures can be estimated reliably. For this purpose, we outlined the formalization of an Approximate PrMF, to demonstrate how the PrMF can be put into practice. For validation, empirical and theoretical studies on representative benchmark problems commonly used in the literature are then presented to demonstrate the characteristics and efficacies of the probabilistic memetic framework. Subsequent comparisons to recent state-of-the-art evolutionary algorithms, memetic algorithms and hybrid evolutionary local demonstrate that the proposed framework converges to good solutions efficiently.

This paper has established the groundwork for the formalization of a quantitative framework for adaptive MA. On the one hand, it can be viewed as a paradigm shift on how adaptation on MAs can be dealt with. From a research point of view, this perception is likely to be more applicable and further research is

warranted. On the other hand, it can be perceived as an alternative on how one can quantitatively characterize the synergy level between evolutionary and individual learning in MAs. This is more likely to be applicable from an implementation point of view.

7 Acknowledgement

The authors wish to thank Professor Xin Yao for his fruitful discussion and constructive comments on an earlier draft of this paper.

8 References

[1] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, *Optimization by Simulated Annealing*, Science, 220(4598):671-680, 1983.

[2] T. Back, F. Hoffmeister and H. Schwefel, *Survey of Evolution Strategies*, editors: R. Belew and L. Booker, Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 2-9, 1991.

[3] L. Davis, *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991.

[4] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolutionary Programs*, Springer-Verlag, New York, 2nd edition, 1994.

[5] A. Auger and N. Hansen, *Reconsidering the Progress Rate Theory for Evolution Strategies in Finite Dimensions*, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2006), pp. 445-452, 2006.

[6] Y. S. Ong and A. J. Keane, *Meta-Lamarckian Learning in Memetic Algorithms*, IEEE Transactions on Evolutionary Computation, 8(2):99-110, 2004.

[7] A. Torn and A. Zilinskas, *Global Optimization*, Volume 350 of Lecture Notes in Computer Science, Springer-Verlag, 1989.

[8] P. Moscato, *On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms*, Publication Report 790, Caltech Concurrent Computation Program, 1989.

[9] C. Houck, J. Joines, and M. Kay, *Utilizing Lamarckian Evolution and the Baldwin Effect in Hybrid Genetic Algorithms*, NCSU-IE Technical Report 96- 01, Meta-Heuristic Research and Applications Group, Department of Industrial Engineering, North Carolina State University, 1996.

[10] A. Vicini and D. Quagliarella, *Airfoil and wing design using hybrid optimization strategies*, American Institute of Aeronautics and Astronautics Journal, 37(5):634-641, 1999.

[11] Y. S. Ong, P. B. Nair and A. J. Keane, *Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling*, American Institute of Aeronautics and Astronautics Journal, 41(4):687-696, 2003.

[12] Z. Z. Zhou, Y. S. Ong, P. B. Nair, A. J. Keane and K. Y. Lum, *Combining Global and Local Surrogate Models to Accelerate Evolutionary Optimization*, IEEE Transactions On Systems, Man and Cybernetics - Part C, 37(1):66-76, 2007.

[13] W. E. Hart, *Adaptive Global Optimization with Local Search*, PhD thesis, University of California, San Diego, May 1994.

[14] K. W. C. Ku, M. W. Mak, and W. C. Siu., *A study of the Lamarckian evolution of recurrent neural networks,* IEEE Transactions on Evolutionary Computation, 4(1):31-42, April 2000.

[15] M. W. S. Land, *Evolutionary algorithms with local search for combinatorial optimization*, Ph. D. Thesis, University of California, San Diego, 1998.

[16] N. K. Bambha, S. S. Bhattacharyya, J. Teich, and E. Zitzler, *Systematic integration of parameterized local search in evolutionary algorithm*, IEEE Transactions on Evolutionary Computation, 8(2):137-155, April 2004.

[17] D. E. Goldberg and S. Voessner, *Optimizing global-local search hybrids,* in GECCO'99, volume 1, pp. 220-228, San Francisco, CA, 1999. Morgan Kaufmann.

[18] A. Sinha, Y. Chen and D.E. Goldberg, *Designing Efficient Genetic and Evolutionary Algorithm Hybrids*, Recent Advances in Memetic Algorithms, Springer, 2005.

[19] X. Yao, *Simulated annealing with extended neighbourhood*, International Journal of Computer Mathematics, 40:169-189, 1999.

[20] N. Krasnogor and J. Smith, *A memetic algorithm with self-adaptive local search: TSP as a case study,* in GECCO-2000: Proceedings of the Genetic and Evolutionary Computation Conference. Morgan Kaufman, pp. 987-994, 2000.

[21] N. Krasnogor and J. Smith, *A tutorial for competent memetic algorithms: model, taxonomy, and design issues*, IEEE Transactions on Evolutionary Computation, 9(5):474- 488, 2005.

[22] Y. S. Ong, M. H. Lim, N. Zhu and K. W. Wong, *Classification of Adaptive Memetic Algorithms: A Comparative Study*, IEEE Transactions On Systems, Man and Cybernetics - Part B, 36(1):141-152 February 2006.

[23] N. Noman., H. Iba, *Accelerating Differential Evolution Using an Adaptive Local Search*, IEEE Transactions on Evolutionary Computation, vol.12, no.1, pp.107-125, 2008.

[24] M. Lozano, F. Herrera, N. Krasnogor, and D. Molina, *Real-coded memetic algorithms with crossover hill-climbing*, Evolutionary Computation Journal, 12(3):273-302, 2004.

[25] Y. S. Ong, N. Krasnogor and H. Ishibuchi, *Special Issue on Memetic Algorithm*, IEEE Transactions on Systems, Man and Cybernetics - Part B, 37(1):2-5, 2007.

[26] Y. S. Ong, M. H. Lim, F. Neri, and H. Ishibuchi, *Special issue on emerging trends in soft computing: memetic algorithms*, Soft Computing-A Fusion of Foundations, Methodologies and Applications, vol. 13, no. 8-9, pp. 1-2, Jul. 2009.

[27] J. E. Smith, *Co-evolving memetic algorithms: A review and progress report*, IEEE Transactions on Systems, Man and Cybernetics, Part B, vol. 37, no. 1, pp. 6–17, Feb. 2007

[28] A. Caponio, G. L. Cascella, F. Neri, N. Salvatore, M. Sumne, *A Fast Adaptive Memetic Algorithm for Online and Offline Control Design of PMSM Drives***,** IEEE Transactions on Systems, Man and Cybernetics, part B, 2007, 37(1):28-41, 2007.

[29] D. Liu , K. C. Tan, C. K. Goh, W. K. Ho, *A Multiobjective Memetic Algorithm Based on Particle Swarm Optimization*, IEEE Transactions on Systems, Man and Cybernetics, part B, 2007, 37(1):42-50, 2007.

[30] M. Tang and X. Yao, *A Memetic Algorithm for VLSI Floorplanning*, IEEE Transactions on Systems, Man, and Cybernetics, Part B, 37(1):62-69, 2007.

[31] J. Tang, M. H. Lim and Y. S. Ong, *Diversity-Adaptive Parallel Memetic Algorithm for Solving Large Scale Combinatorial Optimization Problems*, Soft Computing Journal, 11(9):873-888, Jul. 2007.

[32] Z. Zhu, Y. S. Ong and M. Dash, *Wrapper-Filter Feature Selection Algorithm Using A Memetic Framework*, IEEE Transactions On Systems, Man and Cybernetics - Part B, 37(1):70-76, Feb. 2007.

[33] D. H. Wolpert and W. G. MacReady, *No free lunch theorems for optimization*, IEEE Transactions on Evolutionary Computation, 1(1):67-82, 1996.

[34] G. A. Croes (1958), *A method for solving traveling salesman problems,* Operations Res. 6(6):791-812, 1958.

[35] H. P. Schwefel, *Evolution and Optimum Seeking*, Wiley, 1995.

[36] J. A. Nelder and R. Meade, *A simplex method for function minimization*, Computer Journal, 7(4):308-313, 1965.

[37] M. J. D. Powell, *An efficient method for finding the minimum of a function of several variables without calculating derivatives*, Computer Journal, 7(4):303-307, 1964.

[38] C. Zhu, R. H. Byrd and J. Nocedal, *L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization*, ACM Transactions on Mathematical Software, 23(4):550-560, 1997.

[39] M. D. McKay, R. J. Beckman and W. J. Conover, *A comparison of three methods for selecting values of input variables in the analysis of output from a computer code*, Technometrics, 21(2):239-245, 1979.

[40] R. Storn, Differential Evolution for Continuous Optimization. Available at: http://www.icsi.berkeley.edu/~storn/code.html

[41] H. Beyer and H. Schwefel, *Evolution strategies –A comprehensive introduction*, Natural Computing, 1(1):3-52, 2002.

[42] J. J. Liang, P. N. Suganthan and K. Deb, *Novel Composition Test Functions for Numerical Global Optimization*, IEEE Swarm Intelligence Symposium, pp. 68-75, June 2005.

[43] C. García-Martínez, M. Lozano, *Hybrid Real-Coded Genetic Algorithms with Female and Male Differentiation*, Proceedings of the IEEE Congress on Evolutionary Computation, Edinburgh (UK, 25) 896-903, 2005.

[44] D. Molina, F. Herrera, M. Lozano, *Adaptive Local Search Parameters for Real-Coded Memetic Algorithms*, In Proceeding of the IEEE Congress on Evolutionary Computation, pp. 888-895, 2005.

[45] J. J. Liang and P. N. Suganthan, *Dynamic Multi-Swarm Particle Swarm Optimizer with Local Search*, In Proceeding of the IEEE Congress on Evolutionary Computation, Sept. 2005.

[46] B. Yuan and M. Gallagher, *Experimental Results for the Special Session on Real-Parameter Optimization at CEC 2005: A Simple, Continuous EDA*, In Proceedings of the 2005 Congress on Evolutionary Computation (CEC'05), pp. 1792-1799, 2005.

[47] N. Noman and H. Iba, *Enhancing differential evolution performance with local search for high dimensional function optimization*, In Proceedings of the 2005 Congress on Evolutionary Computation (CEC'05), pp. 967–974, 2005.

[48] A. Auger and N. Hansen, *A restart CMA evolution strategy with increasing population size*, In Proceedings of the 2005 Congress on Evolutionary Computation (CEC'05), pp. 1769-1776, 2005.

[49] X. Yao and Y. Liu, *Fast evolution strategies*, *Control and Cybernetics*. 26(3):467-496, 1997.

[50] X. Yao, Y. Liu and G. Lin, *Evolutionary programming made faster*, IEEE Transactions on Evolutionary Computation, 3(2):82-102, July 1999.

[51] Y. W. Leung, and Y. Wang, *An orthogonal genetic algorithm with quantization for numerical optimization,* IEEE Transactions on Evolutionary Computation, 5(1):41-53, 2001.

[52] J. T. Tsai, T. K. Liu and J. H. Chou, *Hybrid Taguchi-genetic algorithm for global numerical optimization*, IEEE Transactions on Evolutionary Computation, 8(2):365–377, April 2004.

[53] M. H. Lim, S. Gustafson, N. Krasnogor, and Y. S. Ong, *Editorial to the first issue*, Memetic Computing, vol. 1, no. 1, pp. 1-2, Mar. 2009.

[54] I. Paenke, Y. Jin and J. Branke, *Balancing Population-and Individual-Level Adaptation in Changing Environments*, Adaptive Behavior, vol. 17, no. 2 , pp.153-174, 2009

[55] G. Gutin, and D. Karapetyan, *A selection of useful theoretical tools for the design and analysis of optimization heuristics*, Memetic Computing, vol. 1, no. 1, pp. 25-34, Mar. 2009.

[56] S. Hasan, R. Sarker, D. Essam, and D. Cornforth, *Memetic algorithms for solving job-shop scheduling problems*, Memetic Computing, vol. 1, no. 1, pp. 69-83, Mar. 2009.