Integrating Surrogate Evaluation Model and Asynchronous Evolution in Multi-Objective Evolutionary Algorithm for Expensive and Different Evaluation Time

Misaki Kaidan
Ritsumeikan University
1-1-1 Noji-higashi, Kusatsu, Shiga
Japan
is0160xk@ed.ritsumei.ac.jp

Tomohiro Harada
Ritsumeikan University
1-1-1 Noji-higashi, Kusatsu, Shiga
Japan
harada@ci.ritsumei.ac.jp

Ruck Thawonmas
Ritsumeikan University
1-1-1 Noji-higashi, Kusatsu, Shiga
Japan
ruck@ci.ritsumei.ac.jp

ABSTRACT

This paper proposes Extreme Learning Surrogate assisted Asynchronous Multi-Objective Optimization Based on Decomposition (AELMOEA/D) that solves multi-objective optimization problems with expensive and different evaluation time by integrating a surrogate evaluation model and an asynchronous evolution method. Extreme Learning Surrogate assisted Multi-Objective Optimization Based on Decomposition (ELMOEA/D), which is a surrogate-assisted MOEA/D, was proposed to reduce the number of actual evaluations, while asynchronous evolution methods were proposed to reduce the waiting time for evaluation of solutions in a parallel evolutionary algorithm. This paper employs ELMOEA/D as a surrogate assisted EA and introduces an asynchronous manner into it. Our experiment proves that our proposed AELMOEA/D can obtain optimal solutions faster than ELMOEA/D without performance deterioration.

CCS CONCEPTS

• Theory of computation → Evolutionary algorithms; • Theory of computation → Parallel algorithm; • Computing methodologies → Supervised learning by regression;

KEYWORDS

Surrogate evaluation model, Extreme Learning Machine, Asynchronous evolution method, Parallel evaluation, MOEA/D, Multi-objective optimization

ACM Reference format:

M. Kaidan, T. Harada, R. Thawonmas. 2017. Integrating Surrogate Evaluation Model and Asynchronous Evolution in Multi-Objective Evolutionary Algorithm for Expensive and Different Evaluation Time . In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017*, 8 pages.
DOI: 10.1145/3067695.3084215

1 INTRODUCTION

Evolutionary Algorithm (EA) has been applied to many real world problems such as engineering design from the viewpoint of versatility and high search performance [1]. In general EAs, many solution candidates are generated through the optimization process, and their evaluation values are obtained. When applying EAs to real world problems, since the evaluation time of solutions is expensive, enormous calculation time is required to obtain the optimal solution. To tackle this problem, parallel EAs have been proposed by many researchers. However, the evaluation time of solution may differ from each other when the evaluation time of solution is expensive. In such situation, the conventional parallel EAs waste a lot of computational time because they evolve solutions by waiting for the evaluation of all solutions in the population and it is necessary to wait for the slowest evaluation. To overcome these problems caused by the evaluation time of solutions in optimization problems, there are two recent approaches, one is a surrogate assisted EA, while another is an asynchronous evolution.

In the previous researches, several surrogate assisted methods have been proposed such as ParEGO [2], MOEA/D-RBF [3] and ELMOEA/D [4]. A surrogate assisted EA constructs a surrogate evaluation model from already evaluated solutions by using machine learning techniques, and generates promising solutions by some optimizers like EAs or other methods based on a generated surrogate model. For example, in the ELMOEA/D, the surrogate evaluation model is generated by Extreme Learning Machine (ELM) [5], which is a kind of machine learning method. ELMOEA/D generates promising solutions by using Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) [6], which is one of the most powerful multi-objective EA, with a surrogate evaluation model constructed by ELM. Since the evaluation time of a generated surrogate model is extremely shorter than actual evaluation time and actual evaluations are applied only to the promising solutions, a surrogate assisted EA can reduce the computational time of optimization.

On the other hand, an asynchronous EA was proposed to reduce waiting time of computational nodes in a parallel computational environment. Concretely, an asynchronous EA continuously generates a new solution without waiting for other solutions, unlike conventional synchronous approaches need to wait for evaluations of all solutions in a population. Asynchronous EAs can efficiently evolve solutions with different evaluation time since the evolution continues without waiting for other solutions with long evaluation time. For example, asynchronous particle swarm optimization (APSO) [7] and asynchronous differential evolution (ADE) [8] were proposed as methods extending conventional synchronous EAs to asynchronous ones.

Although the effectiveness of these two approaches was revealed, they cannot solve both problems regarding the computation time of solutions. A surrogate assisted EA cannot deal with the difference of the evaluation time in a parallel environment, while an asynchronous EA has limited ability to reduce actual evaluation time. Toward this problem, this research aims to reduce both evaluation time and waiting time in a parallel environment. To achieve this goal, we explore an integration of a surrogate assisted approach with an asynchronous approach. Concretely, in our approach, promising solutions are generated based on a surrogate model, then they are evaluated in a parallel environment and a new solution is generated whenever an actual evaluation of any solution completes as an asynchronous manner. As an application of an integration of surrogate assisted and asynchronous approach, we propose an extreme learning surrogate assisted asynchronous multi-objective evolutionary algorithm based on decomposition (AELMOEA/D), which is a multi-objective EA integrating ELMOEA/D as a surrogate assisted EA with an asynchronous approach. To investigate the effectiveness of as the proposed AELMOEA/D, we conduct the experiment that compares our proposed AELMOEA/D with existing ELMOEA/D by using the ZDT, MOP benchmarks.

The remaining of this paper is organized as follows. Section II shows related works regarding surrogate assisted EA and asynchronous EAs. Section III explains the proposed AELMOEA/D. Section IV conducts the experiment to compare the proposed and the previous methods on MOP benchmarks and shows their result. Finally, Section V concludes this research and shows future work.

2 RELATED WORK

2.1 Multi-objective optimization problem

Multi-objective Optimization Problem (MOP) is a problem of minimizing or maximizing k mutually competing objective functions $\boldsymbol{f}(\mathbf{x})$ (Eq. (1)) [9].

$$\begin{cases} \min_{\mathbf{x}} \boldsymbol{f}(\mathbf{x}) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_k(\mathbf{x})\}^T \\ \text{Subject to: } x_{L,d} \leq x_d \leq x_{U,d} \ (d = 1, \cdots, N_D) \end{cases} \tag{1}$$

$\mathbf{x} = (x_1, \cdots, x_{N_D})$ is a D dimensional solution. $x_{L,d}, x_{U,d}$ are lower or upper bounds of d^{th} variable. $\boldsymbol{f}: \Omega \to R^k$ is a vector having several objective functions and maps the decision variable space Ω to the objective space.

In MOPs, it is difficult to obtain a single solution when each objective function is in a trade-off relationship. Therefore, the goal of MOPs is to achieve the Pareto optimal solution set. The Pareto optimal solution set is defined with the dominance relationship of solutions in MOPs. The definition of the dominance relationship of solutions in the case of minimization of MOP is denoted as follows: When two solutions $\mathbf{x}, \mathbf{y}$ satisfy $\forall j \in \{1, \cdots, k\}: f_j(\mathbf{x}) \leq f_j(\mathbf{y})$ and $\exists m \in \{1, \cdots, k\}: f_m(\mathbf{x}) < f_m(\mathbf{y})$, it is said that $\mathbf{x}$ dominates $\mathbf{y}$ ($\mathbf{x} \prec \mathbf{y}$). The Pareto optimal solution set consists of solutions that are not dominated by any other solutions.

2.2 Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [6]

MOEA/D is a multi-objective evolutionary algorithm that decomposes MOP into many scalar optimization sub-problems based on an aggregation function with the weight vector uniformly distributed in the objective space as shown in Fig. 1. Solutions are arranged on each weight vector direction, and a single objective search is performed locally. For each weight vector, it is possible to search uniform and diverse Pareto solution set on the objective space while maintaining the solution selection pressure by limiting the genetic operation within the neighborhood. The algorithm of MOEA/D is as follows:

1. Initialization
   i. Generate weight vectors $\boldsymbol{\lambda}_i$ spread uniformly ($i = 1, \cdots, N$)
   ii. Explore a neighborhood set $B(i)$ of weight vector $\boldsymbol{\lambda}_i$
   iii. Generate an initial population $\mathbf{x}_1, \cdots, \mathbf{x}_N$
2. Update
   i. Select two parents randomly from $B(i)$
   ii. Generate offspring by using the genetic operators
   iii. Update reference point $\mathbf{z}^*$: ($\mathbf{z}^* = [\min(f_1(\mathbf{x})), \cdots, \min(f_k(\mathbf{x}))]^T$, k is the number of objective functions.)
   iv. Update solutions using an aggregation function $g(\mathbf{x}|\boldsymbol{\lambda}, \mathbf{z}^*)$
3. Stopping Criteria

   If stopping criteria are satisfied, the search is terminated. Otherwise, return to Step 2.

Although several aggregation functions were proposed, this research employs the Penalty Boundary Intersection (PBI) aggregation function. PBI is defined as (Eq. (2)):

$$\begin{cases} g(\mathbf{x}|\mathbf{w}, \mathbf{z}) = d_1 + \theta d_2 \\ d_1 = \dfrac{\|(\mathbf{f}(\mathbf{x}) - \mathbf{z})^T \mathbf{w}\|}{\|\mathbf{w}\|} \\ d_2 = \|\mathbf{f}(\mathbf{x}) - (\mathbf{z} + d_1 \mathbf{w})\|, \end{cases} \tag{2}$$

where θ is the user defined parameter.

2.3 Extreme Learning Surrogate Models in Multi-Objective Optimization Based on Decomposition (ELMOEA/D)

ELMOEA/D is a kind of MOEA with a surrogate evaluation model. ELMOEA/D employs extreme learning machine (ELM) as a surrogate evaluation model and generates promising solutions by using MOEA/D on a constructed ELM model. MOEA/D-DE is

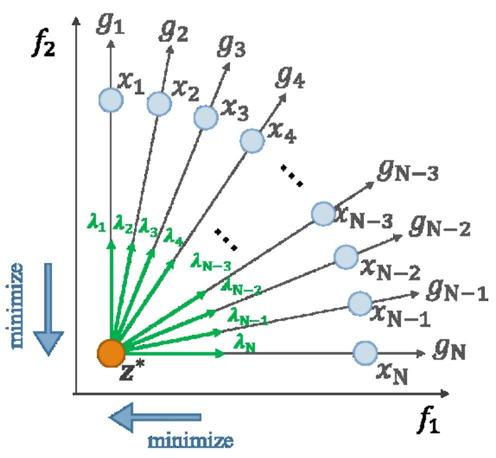
f_2
f_1
g_1 g_2 g_3 g_4 g_{N-3} g_{N-2} g_{N-1} g_N
x_1 x_2 x_3 x_4 x_{N-3} x_{N-2} x_{N-1} x_N
λ_1 λ_2 λ_3 λ_4 λ_{N-3} λ_{N-2} λ_{N-1} λ_N
z^*
minimize
minimize

Figure 1: A concept of MOEA/D

employed in ELMOEA/D, which uses operators from Differential Evolution (DE), in particular DE/rand/1/bin, to generate new solutions based on two control parameters: F and CR. Since the surrogate evaluation model can estimate the evaluation value in shorter time than actual evaluation time and actual evaluation value is calculated only for promising solutions generated by MOEA/D with the surrogate evaluation models, ELMOEA/D significantly reduces the optimization time by reducing the number of actual evaluations.

In their research, ELMOEA/D was compared with conventional surrogate assisted MOEAs (MOEA/D-RBF*ens* and ParEGO) and MOEA/D-DE. From a set of 47 benchmark problems, ELMOEA/D obtained the best results for 30 benchmarks and was statistically equivalent to the best algorithm in other 8 benchmarks. ELMOEA/D showed promising results in MOP benchmarks with twice the decision variables commonly used in the literature. This is because ELM has ability to estimate the evaluation value even in the high dimensional decision variables.

The following subsections explain the detail of ELM and ELMOEA/D.

2.3.1 Extreme Learning Machine (ELM) [5]. ELM is a kind of the machine learning technique. ELM is constructed as a Single Layer Feed Forward Neural Network (SLFN) with K hidden layer neurons and the activation functions $g(\mathbf{w},\mathbf{x},b)$. As the most interesting feature of ELM, the hidden layer parameters are randomly assigned and do not need to be adjusted, while the output weight are only adjusted as follows from N distinct samples of n dimensional inputs $\mathbf{x}_i$ with m dimensional outputs $\mathbf{t}_i$:

Input: N distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$
$(\mathbf{x}_i = [x_{i1}, x_{i2}, \cdots, x_{in}]^T \in \mathbf{R}^n, \mathbf{t}_i = [t_{i1}, t_{i2}, \cdots, t_{im}]^T \in \mathbf{R}^m)$

1. Randomly assign input weight $\mathbf{w}_i$ and bias b_i, (i=1,⋯,K)
2. Calculate the hidden neurons outputs $\mathbf{H}$ by Eq. (3)

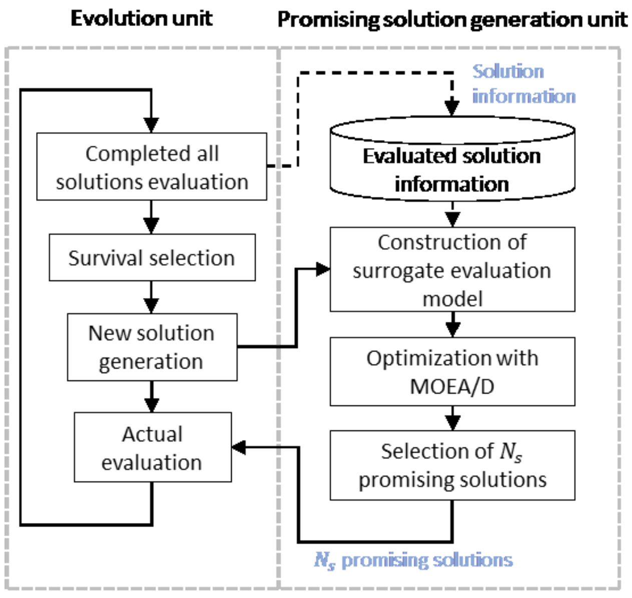
Evolution unit
Promising solution generation unit
Solution information
Completed all solutions evaluation
Evaluated solution information
Survival selection
Construction of surrogate evaluation model
New solution generation
Optimization with MOEA/D
Actual evaluation
Selection of N_s promising solutions
N_s promising solutions

Figure 2: The flow of ELMOEA/D

$$\begin{cases} \mathbf{H} = \{h_{ij}\} \ (i = 1, \cdots, N \text{ and } j = 1, \cdots, K) \\ h_{ij} = g(\mathbf{w}_j, \mathbf{x}_i, b_j) \end{cases} \tag{3}$$

3. Calculate output weight $\boldsymbol{\beta}$ Eq. (4)

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{4}$$

$\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$ [10]. $\mathbf{T} = [\mathbf{t}_1, \cdots, \mathbf{t}_N]^T$ is the matrix of desired outputs.

The advantages of ELM include the following:

- ELM can prevent over-learning since the output weight is the only parameter to be learnt.
- It does not assume the differentiability for the activation function.
- Only one parameter, the number of hidden layers K, should be manually set by the user.
- Iterative processing is not necessary since the output weight is uniquely determined. Therefore, the learning time is fast.

Sigmoid (*SIG*), Gaussian (*GAU*) and Multiquadric (*MQ*) are used mainly as the activation functions of ELM. Three activation functions are defined as follows:

- Sigmoid (SIG):

$$g_{SIG}(\mathbf{w}, \mathbf{x}, b) = \frac{1}{1 + \exp(-(\mathbf{w} \cdot \mathbf{x} + b))} \tag{5}$$

- Gaussian (GAU):

$$g_{GAU}(\mathbf{w}, \mathbf{x}, b) = \exp(-b\|\mathbf{x} - \mathbf{w}\|^2) \tag{6}$$

- Multiquadric (MQ):

$$g_{MQ}(\mathbf{w}, \mathbf{x}, b) = (\|\mathbf{x} - \mathbf{w}\|^2 + b^2)^{1/2} \tag{7}$$

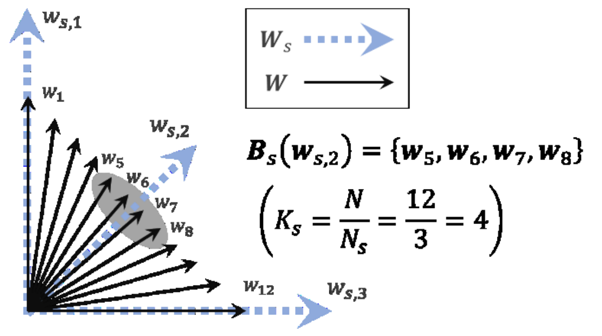
$w_{s,1}$
w_1
$w_{s,2}$
w_5
w_6
w_7
w_8
w_{12}
$w_{s,3}$
W_s
W
$B_s(w_{s,2}) = \{w_5, w_6, w_7, w_8\}$
$\left(K_s = \frac{N}{N_s} = \frac{12}{3} = 4\right)$

Figure 3: Relationship diagram of MOEA/D weight vector and selector set

2.3.2 ELMOEA/D. Figure 2 shows the flow of ELMOEA/D. The left side of this figure shows the process with the actual evaluation, while the right side shows the optimization process on the constructed surrogate model. The algorithm of ELMOEA/D is as follows:

1. Initialization of population, training set, and selector set

   Initial population of MOEA/D (P^0) and ELM's initial training set (P_t^0) are generated by Latin Hypercube Sampling (LHS) [11]. LHS can generate various solution sets within the range of solutions set for each problem. All initial solutions are evaluated with actual evaluation.

   Selector weight sets to select promising solutions are collected. Each selector weight vector $\boldsymbol{w}_s \in \boldsymbol{W}_s$ (Number: N_s) is associated with a subset of the weight vector $\boldsymbol{W}$ of MOEA/D, $\boldsymbol{B}_s(\boldsymbol{w}_{s,n}) = \{\boldsymbol{w}_{n,1}, \cdots, \boldsymbol{w}_{n,K_s}\}$ $(K_s = N/N_s)$. As shown in Fig. 3, K_s neighbor vectors of each selector vector are selected from weight vector $\boldsymbol{W}$ of MOEA/D and are associated in $\boldsymbol{B}_s(\boldsymbol{w}_{s,n})$.

2. Construction of surrogate evaluation model

   The surrogate evaluation model is constructed by learning the output weight of ELM using training set P_t^g. For the activation function of ELM, we select the one with the smallest mean square error (MSE) out of models using sigmoid function, Gaussian function, Multiquadric function. In this procedure, the original expensive function $\boldsymbol{f}$ is transformed into an evaluation function $\boldsymbol{f}_a$ having a low calculation cost.

3. Optimization with MOEA/D on the constructed surrogate model

   The population is optimized up to the maximum number of generations (G generation) with MOEA/D while evaluating with alternative evaluation model $\boldsymbol{f}_a$ constructed in Step 2.

4. Selection of N_s solutions to be actually evaluated (promising solutions)

   From the population P^G optimized up to the Gth generation, promising solutions are selected based on the weight vectors of selector set $\boldsymbol{W}_s$. The fitness of solutions belonging to $\boldsymbol{B}_s(\boldsymbol{w}_{s,n})$ is calculated using aggregate function (PBI is used) based on $\boldsymbol{w}_{n,i} \in \boldsymbol{B}_s(\boldsymbol{w}_{s,n})$. Among them, the best solution is selected as a promising solution belonging to the selector vector $\boldsymbol{w}_{n,i}$. This process is performed for each weight vector set and totally N_s solutions are selected for the actual evaluation.

5. Update population and training set

   Apply actual evaluation to all selected solutions and add them to the population P^{g+1} and select next training set P_t^{g+1} from the evaluated N_s solutions and the previous training set P_t^g.

6. Reconstruction of MOEA/D population

   If the number of the non-dominated solutions is larger than the size of the population, the initial population for MOEA/D is randomly selected from the non-dominant solutions, and if it is not enough, it is generated by LHS.

7. Repeat Step 2 to 6 until the stopping criteria.

2.4 Asynchronous EA

Since conventional parallel EAs generate solution candidates for next generation by evaluating all solutions, they need to wait for completing the slowest evaluation. Therefore, in the case that the optimization problem having different evaluation times, they waste much idling time to wait for the end of the slowest evaluation even if other solutions complete their evaluations quickly. To overcome this problem, some recent researches proposed asynchronous EAs that asynchronously evolve solutions without waiting for the evaluation of other solutions. The general procedure of the asynchronous EA is as follows:

1. Initial population generation
2. Parallel solution evaluation
3. When evaluation of a solution is completed
   A) Survival selection of solution
   B) Generate children from the evaluated solutions
   C) Start evaluation of children

Asynchronous particle swarm optimization (APSO) [7] and asynchronous differential evolution (ADE) [8] are proposed as an extension of conventional synchronous EAs to asynchronous ones. APSO is an asynchronous extension of Particle Swarm Optimization (PSO) [12]. Synchronous PSO obtains all the evaluation values of all particles and then updates the best position g_{best} of the whole group. On the other hand, APSO sequentially updates g_{best} every time the evaluation of each particle is completed. ADE is an asynchronous extension of differential evolution (DE) [13]. In ADE, every time evaluation of each solution is completed, child solutions are generated without waiting for evaluation of other solutions and executes survival selection every time evaluation of each solution is completed.

The advantage of an asynchronous EA is that there is no waiting time because it does not wait for all solution evaluations. However, if the actual evaluation time itself of the solution is expensive, the ability of an asynchronous EA to reduce the computational time for optimization cannot be expected even if the waiting time is reduced.

3 EXTREME LEARNING SURROGATE ASSISTED ASYNCHRONOUS MULTI-OBJECTIVE OPTIMIZATION BASED ON DECOMPOSITION (AELMOEA/D)

3.1 Overview

Although the effectiveness of these two approaches, the surrogate model EAs and asynchronous EAs, was revealed, they cannot solve both problems regarding the computation time of solutions. Concretely, a surrogate assisted EA reduces the number of actual, computationally expensive fitness evaluation by generating promising solutions with a surrogate model, while it cannot deal with the difference of the evaluation time in a parallel environment. On the other hand, an asynchronous EA has limited ability to reduce actual evaluation time.

Toward this problem, this research aims to reduce both evaluation time and waiting time in a parallel environment by integrating a surrogate approach and an asynchronous approach. Toward this aim, we propose Extreme Learning assisted Asynchronous MOEA/D (AELMOEA/D) that introduces the asynchronous evaluation to ELMOEA/D. In AELMOEA/D, solutions are evaluated in a parallel, master-slave computational environment, and a promising solution is asynchronously generated on the ELM surrogate evaluation whenever fitness evaluation of one solution completes.

3.2 The algorithm of AELMOEA/D

To apply an asynchronous approach to ELMOEA/D, AELMOEA/D must select only one solution from solutions optimized by MOEA/D on the surrogate search space, unlike the original ELMOEA/D selects N_s solutions based on the weighted vector of selector set. In this research, the proposed AELMOEA/D selects one selector set in order, and chooses one solution within the selected selector set according to fitness value calculated by an aggregation function.

Figure 4 shows the flow of AELMOEA/D. The procedure of AELMOEA/D is as follows:

1. Initialization of population, training set, and selector set
2. Construction of surrogate evaluation model
3. Optimization with MOEA/D on the constructed surrogate model
4. $i = i + 1 \; mod \; N_s$
5. Selection promising solution(s):
   i. If first generation: Select N_s solutions as same as ELMOEA/D
   ii. Otherwise: The ith (initial value of $i = 1$) selector set $\boldsymbol{B}_s(\boldsymbol{w}_{s,i})$ is selected to choose a current promising solution. The fitness of solutions belonging to $\boldsymbol{B}_s(\boldsymbol{w}_{s,i})$ is calculated using an aggregate function (PBI is used) based on $\boldsymbol{w}_{n,i} \in \boldsymbol{B}_s(\boldsymbol{w}_{s,i})$. Among them, the best solution is selected as a promising solution.
6. Update population and training set
7. Reconstruction of MOEA/D population
8. Repeat Step 2 to 7 until the stopping criteria.

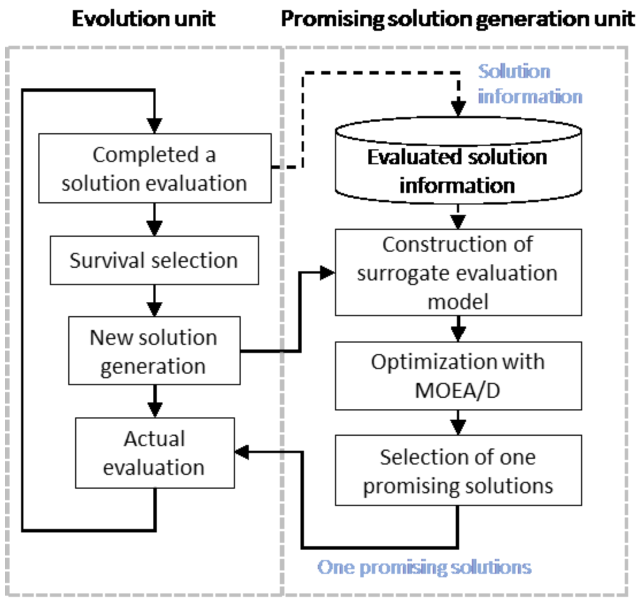
Evolution unit
Promising solution generation unit
Solution information
Completed a solution evaluation
Evaluated solution information
Survival selection
Construction of surrogate evaluation model
New solution generation
Optimization with MOEA/D
Actual evaluation
Selection of one promising solutions
One promising solutions

Figure 4: The flow of AELMOEA/D

In this flow, Steps 1, 2, 3, 6, 7, and 8 are same as the original ELMOEA/D, while Steps 4 and 5 are modified for an asynchronous evolution. In Step 5, since all slave nodes initially idle at first as described in Step 5-i, N_s solutions are selected and sent to slave nodes, while after that, a promising solution is generated whenever one solution evaluation completes as described in Step5-ii.

By simplifying the objective function using the surrogate evaluation model, it is possible to reduce the number of actual evaluations that could not be reduced by an existing asynchronous EA. In addition, it is possible to reduce the waiting time of computational nodes that has occurred in ELMOEA/D by making the evaluation asynchronous.

4 EXPERIMENT

4.1 Overview

To verify the effectiveness of our proposed AELMOEA/D, we conduct the experiment to compare AELMOEA/D with the original ELMOEA/D. The target problem is the ZDT test suite [14] of the two-objective minimization benchmark problems. The design variable is 30 dimensions for of ZDT1 and ZDT2, while 10 dimensions for ZDT3, ZDT4, and ZDT6.

Hypervolume (HV) [15] is used as the evaluation indicator of the obtained Pareto solution set in this experiment. We assess two methods from the viewpoint of the search ability, the computational time, and the convergence speed. The search ability is assessed according to the median HV of 25 independent trials after the maximum number of actual evaluations, while the computation time to complete the maximum number of actual evaluations is compared in both methods. The convergence speed is assessed by comparing the computational time until when the median of HVs in 25 trials reaches to a certain percentage of the maximum HV of each ZDT benchmark. Note that the convergence speed is assessed with 90% for ZDT1, 75% for ZDT2, 85% for ZDT3 and ZDT6, 60% for ZDT4. These percentages are determined according to the result of AELMOEA/D where it achieves HV of given percentage in all 25 trials.

In this experiment, we use the pseudo master-slave parallel computing environment that refers the computational time model proposed in the previous research [16]. Concretely, the fitness evaluation of any solutions is performed in t_p (time units) on any distributed slave nodes, while the sequential computation tasks (i.e., ELM learning and MOEA/D optimization on the surrogate evaluation) are performed in t_s (time units) ($<< \mathrm{t_p}$) on the master node in order to create a new solution. c_v is a degree of the variance of evaluation time, and solution evaluation time is determined according to the normal distribution with the mean t_p and the standard deviation $t_p \times c_v$ if $c_v > 0$.

4.2 Parameter

The following parameters of ELMOEA/D and AELMOEA/D are used in this experiment, which are the same as those in [4]:

- Number of selector weights (N_s): 10
- Training set size: $10N_D$
  (N_D: the number of decision variables of the problem)
- Initial minimum distance (ε): 0.001
- The number of generations (G): 100
- Population size (N): 100
- Neighborhood selection probability (δ): 0.9
- PBI penalty (θ): 5
- DE scaling factor (F): 0.5
- DE crossover rate (CR): 1.0
- Polynomial mutation rate (p_m): $1/N_D$
- Number of hidden neurons (N_H): $2N_D + 1$
- Activation Function (A): $\{SIG,\ MUL,\ GAU\}$
- Regularization parameter (C): $\{2^{-5}, 2^0, 2^5\}$
- Maximum number of actual evaluations (N_E): 2000

while the following settings for the pseudo master-slave parallel environment are used:

- The number of slave nodes: 10 ($= N_s$)
- Mean solution evaluation time (t_p): 1000
- Master node operation time (t_s): 1
- The variance of evaluation time (c_v): {0.02, 0.05, 0.1, 0.2}

4.3 Result

TABLE I shows the median HV value after the maximum actual evaluation number ("HV" column in TABLE I), the calculation time to obtain the defined percentage of the maximum HV of each ZDT benchmark ("Convergence time" column in TABLE I), and the total calculation time to evaluate the maximum number of actual evaluations ("Total time" column in TABLE I) in the proposed AELMOEA/D and the previous ELMOEA/D. Values in parentheses indicate the standard deviation in "HV" and "Total time" columns and the number of trials that reach to the defined percentage of the maximum HV of each ZDT benchmark in 25 independent trials (maximum 25). The bold style in TABLE I indicates that this value is significantly better than another, while "*" marks indicate that the significant difference between two methods is found by the Wilcoxon rank sum test with 5% of significance level (marked with "*") or 1% of significance level (marked with "**").

First, according to "HV" column in TABLE I, it is confirmed that the median values of HV in AELMOEA/D and that in the previous method are not significantly different according to the Wilcoxon rank sum test in ZDT1 with $c_v = 0.1$, ZDT2 with all c_v, ZDT3 with $c_v = 0.05, 0.2$, ZDT4 with $c_v = 0.05, 0.1$, and ZDT6. On the other hand, the median value of the HV value of the previous method is significantly higher than that of the AELMOEA/D with 1% of the significance level in ZDT1 with $c_v = 0.02, 0.05, 0.2$ and ZDT3 with $c_v = 0.02, 0.1$, though in other cases, the median value of the HV value of AELMOEA/D is significantly higher than that of the previous method with 5% of the significance level. Therefore, it is revealed that AELMOEA/D achieves the equivalent performance to the existing method in the same number of actual evaluations, but in some case the asynchronous manner gives negative influence on the search ability of ELMOEA/D.

"Convergence time" column in TABLE I, on the other hand, shows that AELMOEA/D has a shorter calculation time to reach the defined percentage (90% for ZDT1, 75% for ZDT2, 85% for ZDT3 and ZDT6, 60% for ZDT4) of the maximum HV in problems than the previous method regardless of the degree of the variance of solution evaluation time c_v, except for ZDT4. This is because waiting time for solution evaluation was reduced by introducing asynchronous evolution method to ELMOEA/D. In AELMOEA/D, the calculation time to reach 60% is longer in ZDT4, but almost trials achieve 60% of Pareto solution, in contrast to ELMOEA/D that fails some trials. In addition, the median value of the HV value of AELMOEA/D is higher than that of the previous method in ZDT4. Therefore, it can be indicated that the performance has improved in AELMOEA/D.

According to "Total time" column in TABLE I, it is indicated that AELMOEA/D has a shorter computation time to reach the maximum number of actual evaluations in all problems than the previous method, regardless of the degree of the variance of solution evaluation time c_v. In particular, in the previous method, as c_v increases, the calculation time to reach the maximum

TABLE I. The HV value in the maximum actual evaluation number (HV) & the calculation time to obtain 90% (ZDT1), 75% (ZDT2), 85% (ZDT3), 60% (ZDT4) and 50% (ZDT6) of the maximum HV of each ZDT (Convergence time) & the total calculation time up to the maximum actual evaluation number (Total time)

		HV median (standard deviation)		Convergence time median (reach count)		Total time median (standard deviation)	
	c_v	AELMOEA/D	ELMOEA/D	AELMOEA/D	ELMOEA/D	AELMOEA/D	ELMOEA/D
ZDT1	0.02	24.56 (0.08)	**24.59 (0.03)****	**2007.75 (25)**	2074.45 (25)	**170559.95 (120.01)****	176218.97 (159.39)
	0.05	24.53 (0.10)	**24.61 (0.04)****	**2026.33 (25)**	2182.12 (25)	**170684.59 (174.93)****	183965.32 (339.65)
	0.1	24.58 (0.04)	24.60 (0.08)	**2007.42 (25)**	2309.81 (25)	**170407.02 (453.98)****	196874.66 (623.55)
	0.2	24.57 (0.10)	**24.61 (0.02)****	**2058.44 (25)**	2575.96 (25)	**170628.92 (746.54)****	222780.97 (1716.05)
ZDT2	0.02	20.00 (0.00)	20.00 (0.00)	**1991.15 (25)**	2098.73 (25)	**170565.04 (154.62)****	176151.93 (196.66
	0.05	20.00 (0.00)	20.00 (0.00)	**1974.85 (25)**	2185.49 (25)	**170624.83 (188.97)****	184030.81 (431.39)
	0.1	20.00 (0.00)	20.00 (0.00)	**1913.50 (25)**	2375.34 (25)	**170836.79 (363.32)****	197184.29 (1014.93)
	0.2	20.00 (0.00)	20.00 (0.00)	**1883.96 (25)**	2678.46 (25)	**170411.66 (653.60)****	223853.51 (1816.83)
ZDT3	0.02	25.36 (0.61)	**26.57 (0.87)****	**1021.08 (25)**	1042.37 (25)	**190680.98 (147.61)****	196969.25 (198.90)
	0.05	25.64 (0.74)	26.01 (0.99)	**1017.81 (25)**	1090.30 (25)	**190569.56 (287.75)****	205617.71 (527.88)
	0.1	25.44 (0.55)	**26.43 (0.81)****	**1017.60 (25)**	1185.26 (25)	**190455.29 (471.35)****	220437.91 (920.69)
	0.2	25.75 (0.93)	25.98 (0.98)	**1031.76 (25)**	1334.03 (25)	**190729.86 (824.68)****	249036.16 (1401.56)
ZDT4	0.02	**6856.28 (676.40)***	6615.51 (665.13)	45059.43 (25)	**21790.00 (18)**	**190613.47 (162.04)****	196957.36 (166.19)
	0.05	6588.19 (600.26)	6413.05 (710.80)	44426.59 (25)	**38001.15 (20)**	**190705.54 (247.43)****	205562.63 (368.67)
	0.1	6561.99 (525.34)	6408.31 (612.66)	46417.72 (24)	**42823.09 (17)**	**190811.24 (372.43)****	220270.24 (1061.59)
	0.2	**6696.45 (543.81)***	6320.37 (556.94)	**53032.68 (23)**	65529.77 (18)	**190416.22 (926.62)****	249290.02 (1513.72)
ZDT6	0.02	21.49 (0.67)	21.57 (0.71)	**7120.15 (25)**	7244.70 (25)	**190595.69 (105.05)****	196923.39 (158.02)
	0.05	21.76 (0.76)	21.48 (1.21)	7025.62 (25)	**5446.85 (23)**	**190601.11 (211.24)****	205839.56 (432.05)
	0.1	21.61 (0.66)	21.88 (0.80)	**6105.85 (25)**	8211.42 (25)	**190654.57 (462.17)****	220426.76 (852.59)
	0.2	21.67 (0.81)	21.49 (1.18)	**5252.44 (25)**	25535.66 (24)	**190716.22 (765.25)****	249614.64 (1754.47)

From these results, it is revealed that the proposed AELMOEA/D can achieve optimal solutions faster than the previous method without performance deterioration in a parallel computational environment.

5 CONCLUSION

In this research, we aim to reduce both evaluation time and waiting time of optimization problems that deal with solutions having expensive and different evaluation time in EAs in a parallel computational environment. To achieve this goal, we proposed AELMOEA/D, which is a MOEA integrating ELMOEA/D that is a surrogate assisted EA to reduce evaluation time of expensive solutions and the asynchronous approach that reduces waiting time in the situation to optimize solutions with the large variance of evaluation time.

From the result of the comparative experiment of the proposed AELMOEA/D with the previous ELMOEA/D in the pseudo-parallel computational environment, it is indicated that, in multi-objective optimization problems, the HV value of solutions obtained by AELMOEA/D at the maximum number of actual evaluations is mostly equal to that of the previous method. AELMOEA/D has a shorter computation time to reach the maximum actual evaluation number in almost problems than the previous method. In addition, in the proposed method, the computational time to reach to a certain percentage of the maximum HV is shorter than the previous method. This is because waiting time for solution evaluations was reduced by introducing an asynchronous evolution method to ELMOEA/D. Therefore, our experiment proves that AELMOEA/D can achieve optimal solutions faster than the previous method without performance deterioration in a parallel computational environment.

In this research, we select the method of choosing promising solutions in an asynchronous order, but there is a possibility that solution accuracy and convergence speed may be improved depending on the method of choosing a promising solution. From this viewpoint, we will explore other improved selection methods of a promising solution in an asynchronous manner and verify their effectiveness. We will not only tackle this issue, but also explore the timing of ELM learning in an asynchronous evolution and the use of recent MOEA methods like NSGA-III [17] or MEMO [18]. Additionally, we will verify the effectiveness of the proposed method with other recent benchmarks like WFG test suite [19].

REFERENCES

[1] M. Ueno, S. Usui, H. Tanaka, A. Watanabe. Technological overview of the next generation Shinkansen high-speed train Series N700. 2008.
[2] J. Knowles, E. Hughes. Multiobjective optimization on a budget of 250

evaluations. Evolutionary Multi-Criterion Optimization, vol.3410, Springer, Berlin, Heidelberg, 2005, pp.176-190.

[3] S. Z. Martínez, C. A. C. Coello. MOEA/D assisted by RBF networks for expensive multi-objective optimization problems. Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation Conference, 2013, pp.1405-1412.

[4] L. M. Pavelskia, M. R. Delgado, C. P. Almeida, R. A. Gonalves, S. M. Venske. "Extreme Learning Surrogate Models in Multi-objective Optimization based on Decomposition." Neurocomputing 180, 2016, pp. 55-67.

[5] G. Huang, Q. Zhu and C. Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on. Vol. 2. IEEE, 2004, pp. 985-990.

[6] Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on evolutionary computation 11.6, 2007, pp.712-731.

[7] A. Lewis, S. Mostaghim, and I. Scriven, Asynchronous multi-objective optimisation in unreliable distributed environments, in Biologically- Inspired Optimisation Methods, ser. Studies in Computational Intelligence, A. Lewis, S. Mostaghim, and M. Randall, Eds. Springer Berlin Heidelberg, vol.210, 2009, pp.51-78.

[8] M. Depolli, R. Trobec, and B. Filipic. Asynchronous master-slave parallelization of differential evolution for multi-objective optimization. EVOLUTIONARY COMPUTATION, vol.21, no.2, 2013, pp.261-291.

[9] K. Deb. Multi-objective optimization using evolutionary algorithms. John Wiley & Sons on. Vol. 16, 2001.

[10] D. Serre. Matrices: Theory and Applications, Springer, New York, 2002.

[11] 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 42.1, 2000, pp.55-61.

[12] K. James. Particle swarm optimization. Encyclopedia of machine learning. Springer US, 2011. pp.760-766.

[13] S. Rainer and K. Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization 11.4, 1997, pp.341-359.

[14] K. Deb and T. Meyarivan. Constrained test problems for multi-objective evolutionary optimization. KanGAL report 200005, Indian Institute of Technology, Kanpur, India, 2000.

[15] J. Knowles, L. Thiele, and E. Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. In TIK Report 214, Computer Engineering and Networks Laboratory(TIK), ETH Zurich, 2006.

[16] A. Zvoianu, E. Lughofer, W. Koppelsttter, G. Weidenholzer, W. Amrhein, E. P. Klement. Performance comparison of generational and steady-state asynchronous multiobjective evolutionary algorithms for computationally-intensive problems. Knowledge-Based Systems Volume 87, 2015, pp.47-60.

[17] K. Deb and H. Jain. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints, IEEE Transactions on Evolutionary Computation, Volume: 18, Issue: 4, Aug. 2014.

[18] C. C. Tutum and K. Deb, A Multimodal Approach for Evolutionary Multi-objective Optimization (MEMO): Proof-of-Principle Results, 8th International Conference, EMO 2015, Guimaraes, Portugal, March 29 --April 1, 2015. Proceedings, Part I, 2015.

[19] S. Huband, L. Barone, L. While, and P. Hingston. 2005. A Scal- able Multi-objective Test Problem Toolkit. Springer Berlin Heidelberg, Berlin, Hei-delberg, 280–295.