Y. Tanigaki, Y. Nojima, and H. Ishibuchi, "Meta-optimization based multi-objective test problem generation using WFG toolkit," *Proc. of 2016 IEEE Congress on Evolutionary Computation (Accepted).*

Meta-Optimization based Multi-Objective Test Problem Generation using WFG Toolkit

Yuki Tanigaki, Yusuke Nojima, and Hisao Ishibuchi
Department of Computer Science and Intelligent Systems
Graduate School of Engineering, Osaka Prefecture University
Sakai, Osaka 599-8531, Japan
{yuki.tanigaki@ci., nojima@, hisaoi@}cs.osakafu-u.ac.jp

***Abstract*—Selecting a proper set of test problems is essential for fair performance comparison of evolutionary multi-objective optimization (EMO) algorithms. This is because the comparison results strongly depend on the choice of test problems. Test problems are also very important for examining the behavior of each algorithm. In general, it is advisable to prepare a set of various test problems including both easy and difficult ones for each algorithm. Our idea is to use a meta-optimization technique for generating such a set of test problems. More specifically, we use a two-level meta-optimization model. In the upper level, test problems are optimized. That is, test problems are handled as solutions. In the lower level, each test problem is evaluated using multiple EMO algorithms. The point of our idea is high flexibility in the definition of an objective function in the upper level. For example, when we want to design a difficult test problem only for a particular EMO algorithm, the minimization of its relative performance can be used as an objective function. By maximizing its relative performance, we can also design an easy test problem only for that algorithm. By generating both easy and difficult problems for each algorithm in this manner, we can prepare an appropriate test problem set for fair performance comparison. Through computational experiments, we demonstrate that we can generate a wide variety of test problems, each of which is difficult for a different type of EMO algorithms.**

Keywords*—evolutionary multi-objective optimization (EMO); meta-optimization; problem generation; WFG toolkit.

I. INTRODUCTION

In the last three decades, a number of evolutionary multi-objective optimization (EMO) algorithms have been proposed [1]-[6]. Their successful applications have also been reported [7]. Since those EMO algorithms have different search mechanisms, it is clear that they have their own advantages and disadvantages. However, it is not easy to correctly identify and analyze them from performance comparison results for only a small number of frequently-used test problems.

In order to investigate the performance of algorithms in more detail, it is required to select appropriate test problems. In general, it is advisable to prepare a set of various test problems including both easy and difficult ones for each algorithm. In addition, it is desirable to use the test problems that can make a remarkable difference in the performance among algorithms. For example, even if no difference is observed in the comparison between algorithms A and B on a test problem X, it may happen that the performance of either one of the algorithms is severely deteriorated on the other test problem Y. In such a case, the comparison using the test problem Y seems to be more beneficial for examining the behavior of each algorithm.

For the comprehensive comparison of algorithms, test problem suits including different characteristics are proposed such as DTLZ test problems [8], MOP test instances for CEC 2009 MOEA competition [9] etc. However, these test problem suits include only a limited range of multi-objective optimization problems. One straightforward idea to cope with this problem is to generate the appropriate test problems in an automatic way. In this paper, we generate new test problems based on meta-optimization. Meta-optimization is a widely-used automated approach especially for solving algorithm configuration [10]-[12]. In the meta-optimization for the algorithm configuration, a parameter setting of an algorithm is regarded as an optimization problem and a so-called *tuner* optimizes the performance of a target algorithm (i.e., finds appropriate parameter setting). Meta-optimization is also referenced as off-line optimization or hyper-heuristics [13].

In our test problem generation, a two-level meta-optimization model is utilized. In the upper level, test problems are optimized. That is, test problems are handled as solutions by the tuner. In the lower level, each test problem is evaluated using multiple EMO algorithms. In more detail, obtained results from the execution of EMO algorithms are employed to fitness evaluation of the test problem. The definition of an objective function in the upper level (i.e., the tuner) can be changed according to the optimal feature of the test problem. For example, when we want to design a difficult test problem only for a particular EMO algorithm, the minimization of its relative performance can be used as an objective function.

In the computational experiments, we demonstrate that we can generate a wide variety of test problems, each of which is difficult for a different type of EMO algorithms. The functions defined by the WFG toolkit [14] are combined for the test problem construction. Since the characteristics of the obtained test problems can be determined by these functions, we can observe the relationship between the characteristics of obtained test problems and their own target EMO algorithm.

The rest of this paper is organized as follows. We describe a basic structure of meta-optimization in Section II. Next, the problem construction based on the WFG toolkit is explained in Section III. The details of our meta-optimization based test

problem generation are shown in Section IV and its efficiency is discussed through computational experiments in Section V. Finally, we conclude the paper in Section VI.

II. Overview of Meta-Optimization

In recent years, many high-performance EMO algorithms have been proposed with numerous parameters which control their behavior. Search performance of these algorithms may greatly depend on the parameter setting. In addition, it is often the case that there are different optimal parameters for each test problem to be applied. When the user uses these algorithms, it is required to select the appropriate parameters for a given problem. However, in order to properly adjust the parameters, the user needs adequate knowledge of both problems and algorithms. To free the user from such difficulty, a study of the automatic algorithm configuration has been actively carried out. The automatic methods for optimizing a performance of the target algorithm are referred to as meta-optimization.

In the meta-optimization method, a parameter setting of the algorithm is regarded as one of the optimization problems. The goal of the meta-optimization method is to find the optimal parameter setting for a given range of problem instances. We illustrate an outline of parameter tuning using meta-optimization in Fig. 1.

Two different optimization are performed in meta-optimization. One is the execution of the target algorithm corresponding to the configuration scenario in Fig. 1. This part can be considered as a fitness evaluation of the parameter tuner. The other search operation is the selection of an appropriate parameter setting by the parameter tuner. Based on the result of the execution of the target algorithm, the parameter tuner determines which parameters to be applied. When the target algorithm is assumed to be an EMO algorithm, the quality indicator such as hypervolume [15] and IGD [16] can be used for the fitness calculation of the parameter tuner.

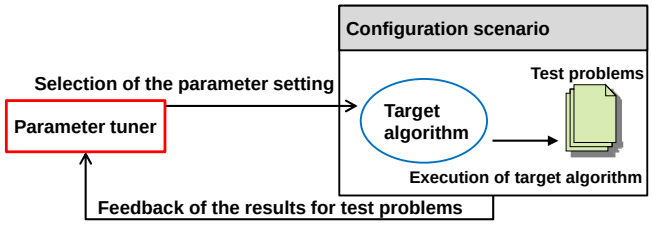


Fig. 1. An outline of the meta-optimization for the algorithm configuration.

The framework of meta-optimization is also used for other purposes as well as parameter settings. For example, a meta-optimization method selects or combines multiple search techniques and designs the efficient search structures in the literature [17], [18]. In this paper, we generate new test problems based on meta-optimization. For this purpose, test problems are created instead of the selection of parameter settings in Fig. 1. That is, the tuner (not "parameter" tuner any more) decides a next candidate test problem based on the result of the execution of algorithms. Unlike the parameter tuning, existing quality indicators are not directly used for the objective function of the tuner. Let us denote two algorithms A and B for an example of fitness calculation. The tuner tries to generate a difficult test problem for algorithm A. In such a case, its relative performance such as E(A) / E(B) can be used for the evaluation method. Where E(A) is a value of any metric to evaluate the execution results of the algorithm A. When the larger value of E(·) means better, the minimization of E(A) / E(B) can be used as an objective function.

One additional important issue of test problem generation is how to create candidate test problems. In this paper, we combine several functions proposed in the WFG toolkit. The detail of the WFG toolkit is explained in the next section.

III. WFG Toolkit

The WFG toolkit [14] is a toolkit for creating scalable multi-objective minimization test problems proposed by Huband et al. It includes several functions determining the characteristics of multi-objective problems. The nine problems (WFG1-WFG9) using this toolkit have been proposed and widely used. All M objective minimization problems constructed by the WFG toolkit conform to the following format:

$$f_{m=1:M}(\mathbf{x}) = x_M + S_m h_m(x_1, ..., x_{M-1}), \quad (1)$$

where h_m is a shape function that determines the shape of the Pareto front, S_m is a parameter to change the scaling of each objective function, and $\mathbf{x} = (x_1, x_2, ..., x_M)^T$ is referred to the underlying variables. The underlying variables are calculated from decision variables $\mathbf{z} = (z_1, z_2, ..., z_n)^T$ by applying various transformation functions. The transformation calculation is shown in (2) and (3). The features of the problem (e.g., multi-modal, dependency among decision variables) are determined by the transformation functions. The number of decision variables is set to $k + l = n$. The first k variables are position-related and the last l variables are distance-related variables.

$$\mathbf{x} = (\max(t_M^p, A_1)(t_1^p - 0.5) + 0.5, ..., \max(t_M^p, A_{M-1})(t_{M-1}^p - 0.5) + 0.5, t_M^p), \quad (2)$$

$$\mathbf{t}^p = (t_1^p, ..., t_M^p)^T \leftarrow \mathbf{t}^{p-1} \leftarrow ... \leftarrow \mathbf{t}^1 \leftarrow \mathbf{z}. \quad (3)$$

The transformation functions are classified roughly three classes. All transformation functions prepared in WFG toolkit are summarized as follows.

Bias functions:
Polynomial function, Flat function,
Parameter Dependent function.

Shift functions:
Linear function, Multi-modal function, Deceptive function.

Reduction functions:
Weighted sum function, Non-separable function.

The bias functions give a bias against the fitness landscape of the decision variables. By using a polynomial function, the location of a randomly generated solution set is pressed into the specific area of the objective space. With the flat function, the fitness landscape of the decision variables does not change in the specific interval (i.e., decision variables have the flat

region). With the parameter dependent function, the fitness landscape of the decision variables is changed depending on the other decision variables. The shift functions change the optimal value of the decision variables. Constructed test problems become uni-modal problems with the linear function and multi-modal problems with the multi-modal function. With the deceptive function, constructed test problems can be considered as deceptive problems whose objective values significantly deteriorate around the optimal values of decision variables. The reduction functions are used for reducing the arbitrary number n of decision variables to the number of underlying variables M. Decision variables have dependencies on each other using the non-separable function, whereas there is no dependency with the weighted sum function.

The shape functions h_m determine the shape of the Pareto front. The general shape of the Pareto front can be freely selected among Linear, Convex, and Concave functions. In addition, there are combinational shape functions which determine disconnected or mixed feature of the Pareto front. In other words, the total number of the shapes of the Pareto front is nine such as Linear, Convex, Concave, Linear-mixed, Convex-mixed, Concave-mixed, Linear-disc, Convex-disc, and Concave-disc. By selecting one among these shapes, it is possible to change the shape of the Pareto front of the problems constructed.

IV. Problem Generation Based on Meta-Optimization

In this section, we describe the detail of the test problem generation using a meta-optimization technique. For generation of the test problem, functions determining the features of the WFG toolkit are combined. Corresponding to the selected functions, the characteristics of the obtained problem are clearly shown. The characteristics of the problem used in meta-optimization are summarized in Table I. In some of the transformation functions, control parameters are employed to determine the detail feature of their characteristic. In this paper, the same settings are used as in WFG 1-9. The effect of these control parameters is discussed in Section V-C.

To construct a test problem, one function is chosen from each category of the characteristics. For example, when the problem includes linear function, polynomial function, and weighted sum function, the constructed test problem has a polynomial bias and a uni-modal property. When all S_m are fixed to 1 and the concave function is used for h_m, the problem has concave Pareto front and each objective function is the same scale. The size of the meta-optimization search space is $3 \times 5 \times 3 \times 2 \times 9 = 810$.

Next, we describe the detail of fitness evaluation in the meta-optimization. In the meta-optimization of this paper, an EMO algorithm and its competitor algorithms are utilized for a fitness evaluation. The tuner tries to generate test problems so that the target algorithm is disadvantageous or advantageous among the algorithms. Let us assume the tuner tries to generate a disadvantageous test problem for the target algorithm A among its competitor algorithms B, C, D. That is, the obtained test problem is relatively a difficult problem for A. In order to calculate a fitness value from the results of the execution of algorithms, hypervolume is used as the quality indicator. The fitness value in the meta-optimization is determined as follows.

$$\text{Minimize} \quad \mathit{fitness} = \mathit{Rank}(\mathrm{A}) + \mathit{HVn}(\mathrm{A}), \tag{4}$$

where $\mathit{Rank}(\mathrm{A})$ is the ranking of A among its competitor algorithms B, C, D. For example, when $\mathit{HV}(\mathrm{B}) > \mathit{HV}(\mathrm{A}) > \mathit{HV}(\mathrm{C}) > \mathit{HV}(\mathrm{D})$ holds, where $\mathit{HV}(\mathrm{A})$ means the hypervolume value obtained from algorithm A, $\mathit{Rank}(\mathrm{A})$ is calculated as 3. The second measure $\mathit{HVn}(\mathrm{A})$ means a normalized hypervolume as follows.

$$\mathit{HVn}(\mathrm{A}) = \frac{\mathit{HV}(\mathrm{A})}{\mathit{HV}(\mathrm{A}) + \mathit{HV}(\mathrm{B}) + \mathit{HV}(\mathrm{C}) + \mathit{HV}(\mathrm{D})}. \tag{5}$$

It should be noted that this fitness evaluation is noisy because a different hypervolume value is obtained from a different run of the same algorithm. The number of executions to evaluate its performance is an important issue for the efficient optimization. That is, when the fitness evaluation is based on the outcomes of single runs the search space of the meta-optimization could be very noisy. On the other hand, enormous calculation times are required to calculate the fitness evaluation on the basis of large number of executions.

TABLE I. Problem Features and Related Functions.

Category of the characteristics	**Characteristics**	**Function definition**
Modality	Uni-modal	Linear function in WFG1
	Multi-modal	Multi-modal function in WFG4
	Deceptive	Deceptive function in WFG5
Bias	--	Bias function is Unused
	Polynomial	Polynomial function in WFG1
	Flat	Flat function in WFG1
	PositionToDsitance	Parameter Dependent function in WFG7
	DsitanceToPosition	Parameter Dependent function in WFG8
Scaling	Same	$S_m = 1$
	Double	$S_m = 2m$
	Exponential	$S_m = 10^m$
Separability	Separable	Weighted sum function in WFG1
	Non-separable	Non-separable function in WFG6
Geometry of the Pareto front (h_m)	Linear	$\text{linear}_{1:M}$
	Convex	$\text{convex}_{1:M}$
	Concave	$\text{concave}_{1:M}$
	Linear-mixed	$\text{linear}_{1:M-1}$ and mixed_M
	Convex-mixed	$\text{convex}_{1:M-1}$ and mixed_M
	Concave-mixed	$\text{concave}_{1:M-1}$ and mixed_M
	Linear-disc	$\text{linear}_{1:M-1}$ and disc_M
	Convex-disc	$\text{convex}_{1:M-1}$ and disc_M
	Concave-disc	$\text{concave}_{1:M-1}$ and disc_M

In order to handle this problem, some tuners have mechanisms that adaptively estimate the number of runs used for the fitness evaluation for each test problem. In this paper, we use SMAC [10] as the tuner of the meta-optimization. SMAC is one of the most famous approaches for algorithm configuration. SMAC has a mechanism that automatically selects how many runs are needed for the fitness evaluation. As already mentioned in Section II, existing algorithm configuration tuners can be used for the tuner in our automatic problem generation method. We show an outline of our problem generation in Fig. 2.

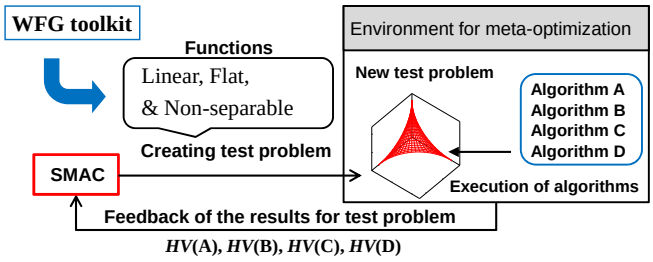


Fig. 2. An outline of the meta-optimization for the test problem generation.

V. Computational Experiments

In this section, we demonstrate the meta-optimization based problem generation through computational experiments on the following six well-known EMO algorithms.

NSGA-II [1]:
NSGA-II is one of the most frequently-used Pareto dominance-based EMO algorithms. Its fitness evaluation is based on a rank assignment mechanism and a secondary measure for diversity maintenance called "crowding distance".

MOGLS [2]:
MOGLS is the hybrid version of NSGA-II with weighted sum-based local search. The local search is utilized to enhance the search performance of NSGA-II.

NSGA-III [3]:
NSGA-III is also a Pareto dominance-based EMO algorithm. The secondary measure of NSGA-II (i.e., crowding distance) is replaced with the solution assignment to reference points in the normalized objective space. This modification enhances the diversity handling performance of NSGA-II.

MOEA/D [4]:
MOEA/D is an efficient scalarizing function-based EMO algorithm. A multi-objective problem is decomposed into a number of single-objective problems. Each single-objective problem is defined by the scalarizing function with a different weight vector. A single solution is used for the optimization on each single-objective problem and solutions are handled in a collaborative manner.

MOEA/D-DE [5]:
MOEA/D-DE is a variant of MOEA/D that replaces the recombination schemes in MOEA/D with the differential evolution (DE) operator.

SMPSO [6]:
SMPSO is an EMO algorithm based on the particle swarm optimization. For the maintenance of an external archive, the rank assignment mechanism is utilized.

A. Algorithm Comparison on WFG1-WFG8

First of all, the performance of the six algorithms is compared on WFG1-WFG8 with 2 and 6 objectives. The number of position-related variables and distance-related variables is specified as $k = 18$ and $l = 5$. We use the following parameter specifications in all the six EMO algorithms:

Termination condition: 500 generations,
Crossover probability: 1.0,
Mutation probability: $1/n$ (n: # of decision variables),
Population size: 100 for 2 objectives, 120 for 6 objectives.

These specifications of the population size are based on the combinatorial nature of the possible number of weight vectors in MOEA/D. In our study, the simulated binary crossover [19] (SBX) and the polynomial mutation [20] (PM) are used for NSGA-II, MOGLS, NSGA-III, and MOEA/D. In MOEA/D-DE and SMPSO, their own reproduction methods are used. Their distribution indices are specified as 15 for SBX and 20 for PM. In MOEA/D and MOEA/D-DE, we use the Tchebycheff function for scalarizing function. Other control parameters in each algorithm are used the same settings as in their original papers.

We show the average hypervolume values over 20 runs in Tables II and III. We used the worst values of each objective function as the reference point for the hypervolume calculation. In Table IV, the characteristics of WFG1-WFG8 are summarized. In Table II and Table III, we can observe the performances of MOEA/D are deteriorated in most of all two-objective WFG problems whereas MOEA/D, MOEA/D-DE, and SMPSO are deteriorated in six-objective problems. However, it is difficult to discuss their advantages and disadvantages related to the characteristics of WFG1-WFG8. As summarized in Table IV, there are many problems which have the same characteristics. For example, the modalities of WFG1-WFG3 and WFG6-WFG8 are uni-modal. The shapes of the Pareto front are concave in WFG4-WFG8. Especially, the same setting (i.e., double) is used for the scaling among objective functions in WFG1-WFG8.

B. Test Problems Generated by Meta-optimization

Next, we generate two-objective and six-objective problems that have the disadvantageous characteristics for a particular target algorithm by meta-optimization. That is, the obtained hypervolume value of a target algorithm is relatively deteriorated in the generated test problem. For its competitor algorithms, the other five algorithms are used.

Termination condition of SMAC is set to 1000 solution evaluations. The number of solution evaluations is larger than the size of the meta-optimization solution space (i.e., 810). This is because the fitness evaluation based on the single algorithm runs is treated as a single solution evaluation in SMAC. In order to decrease negative effects of the noisy fitness evaluation, each test problem is re-evaluated by SMAC mechanism.

TABLE II. OBTAINED HYPERVOLUME ON WFG1-WFG8 WITH 2 OBJECTIVES. THE WORST RESULTS ON EACH PROBLEM ARE SHOWN IN BOLD.

Problem	NSGA-II	MOGLS	NSGA-III	MOEA/D	MOEA/D-DE	SMPSO
WFG1	7.21	6.16	7.34	**4.61**	9.64	10.45
WFG2	9.25	9.26	9.26	**9.09**	9.25	11.45
WFG3	10.42	10.82	10.29	**10.14**	10.46	10.94
WFG4	**7.82**	8.53	7.83	8.29	8.26	8.54
WFG5	7.69	8.07	**7.68**	7.77	7.82	8.04
WFG6	7.73	7.95	7.71	**7.38**	8.18	8.52
WFG7	6.41	8.1	**6.31**	6.44	7.08	8.67
WFG8	6.35	7.51	6.18	**5.72**	6.88	8.36

TABLE III. OBTAINED HYPERVOLUME ON WFG1-WFG8 WITH 6 OBJECTIVES. THE WORST RESULTS ON EACH PROBLEM ARE SHOWN IN BOLD.

Problem	NSGA-II	MOGLS	NSGA-III	MOEA/D	MOEA/D-DE	SMPSO
WFG1	1.16E+05	6.41E+04	1.03E+05	1.11E+05	1.25E+05	**5.00E+04**
WFG2	1.34E+05	1.09E+05	1.13E+05	**1.08E+05**	1.19E+05	1.33E+05
WFG3	9.13E+04	9.14E+04	8.57E+04	7.29E+04	**7.22E+04**	9.05E+04
WFG4	6.31E+04	8.30E+04	1.02E+05	**6.13E+04**	7.68E+04	8.81E+04
WFG5	6.99E+04	8.39E+04	1.06E+05	7.18E+04	**5.42E+04**	5.91E+04
WFG6	7.88E+04	8.21E+04	1.06E+05	6.98E+04	**5.82E+04**	8.08E+04
WFG7	7.38E+04	8.80E+04	1.03E+05	5.97E+04	**4.85E+04**	9.37E+04
WFG8	6.49E+04	6.80E+04	8.61E+04	**4.08E+04**	4.54E+04	7.82E+04

TABLE IV. THE CHARACTERISTICS OF WFG1-WFG8.

Problem	Modality	Bias	Separability	Scaling	Geometry
WFG1	Uni-modal	Polynomial, Flat	Separable	Double	Convex-mixed
WFG2	Uni-modal	-	Non-separable	Double	Convex-disc
WFG3	Uni-modal	-	Non-separable	Double	Linear, Degenerate
WFG4	Multi-modal	-	Separable	Double	Concave
WFG5	Deceptive	-	Separable	Double	Concave
WFG6	Uni-modal	-	Non-separable	Double	Concave
WFG7	Uni-modal	PositionToDsitance	Separable	Double	Concave
WFG8	Uni-modal	DsitanceToPosition	Separable	Double	Concave

The characteristics of obtained test problems are showed in Tables V, VI. Obtained hypervolume values of the six EMO algorithms on each test problem are also summarized in Tables VII, VIII. The obtained test problem is referred to as $TargetAlgorithm_{Min}$ (e.g., NSGA-II_{Min} means an obtained test problem that has the disadvantageous characteristics for NSGA-II). Each test problem shown in Tables V-VIII is the best results (i.e., best fitness value) over 5 times of the meta-optimization trials.

As shown in Tables V, VI, a wide variety of test problems are generated. Although the parameter settings of each function for the test problem generation are picked from WFG1-WFG8, the obtained test problems have totally different characteristics from WFG1-WFG8. In Tables VII, VIII, the worst results on each test problem are highlighted in bold. In Tables VII, VIII, we can observe that the worst results on each test problem are obtained by its target algorithm (e.g., NSGA-II shows the worst results on NSGA-II_{Min}). These results show that the test problems which are difficult for their target algorithms are successfully obtained through meta-optimization.

In Tables V, VI, the characteristics obtained by four or five trials are highlighted by gray. When we focus on the category of characteristics, we can say that the same disadvantageous characteristics are obtained for the bias, separability, and scaling functions in almost all trials. This observation suggests that these three characteristics play the principal role for the comparison among the six EMO algorithms.

In Figs. 3-6, we show the obtained results of the six algorithms on NSGA-III_{Min}, MOEA/D_{Min}, MOEA/D-DE_{Min} and $SMPSO_{Min}$. In Figs. 3-6, each objective values are

normalized using the worst Pareto-optimal objective values. The randomly generated solutions are also shown in black circles.

From the comparison of Figs. 3, 4 and Figs. 5, 6, we can observe that the diversity of the randomly generated initial populations has a large influence on the performance of the EMO algorithms. As we explained in Section III, a randomly generated solution set is pressed into the specific area of the objective space of the problems including the polynomial bias. Both NSGA-III$_{Min}$ and MOEA/D$_{Min}$ include polynomial bias functions. In Fig. 4, the performance of NSGA-II, MOGLS, NSGA-III and MOEA/D are deteriorated. The SBX crossover and the polynomial mutation are used for the reproduction in these EMO algorithms. This may suggest the efficiency of the reproduction operators of MOEA/D-DE and SMPSO (i.e., the differential evolution operator and the particle swarm opmitization-based recombination mechanism).

In Fig. 5, the obtained results of the six algorithms on MOEA/D-DE$_{Min}$ are shown. From Fig. 5-(d) and Fig. 5-(e), the obtained solutions from scalarizing function-based EMO algorithm are biased in the first objectives. This is because each objective values are directory used in the scalarizing function. As shown in Table V, the exponential function is used for the scaling function of MOEA/D-DE$_{Min}$. That is, there is the large difference of the scaling among objective functions. The differences of scaling of MOEA/D$_{Min}$ and MOEA/D-DE$_{Min}$ are double or exponential in both two- and six-objectives. In order to obtain uniformity distributed solutions, some normalization technique handling the difference of the scaling among objective functions is needed.

TABLE V. THE CHARACTERISTICS OF GENERATED TWO-OBJECTIVE PROBLEMS.

Problem	Modality	Bias	Separability	Scaling	Geometry
NSGA-II$_{Min}$	Multi-modal	Flat	Separable	Double	Concave
MOGLS$_{Min}$	Multi-modal	PositionToDistance	Non-separable	Double	Concave
NSGA-III$_{Min}$	Multi-modal	Polynomial	Non-separable	Exponential	Convex-mixed
MOEA/D$_{Min}$	Deceptive	Polynomial	Non-separable	Exponential	Convex-disc
MOEA/D-DE$_{Min}$	Multi-modal	PositionToDistance	Separable	Exponential	Concave
SMPSO$_{Min}$	Multi-modal	PositionToDistance	Separable	Same	Convex

TABLE VI. THE CHARACTERISTICS OF GENERATED SIX-OBJECTIVE PROBLEMS.

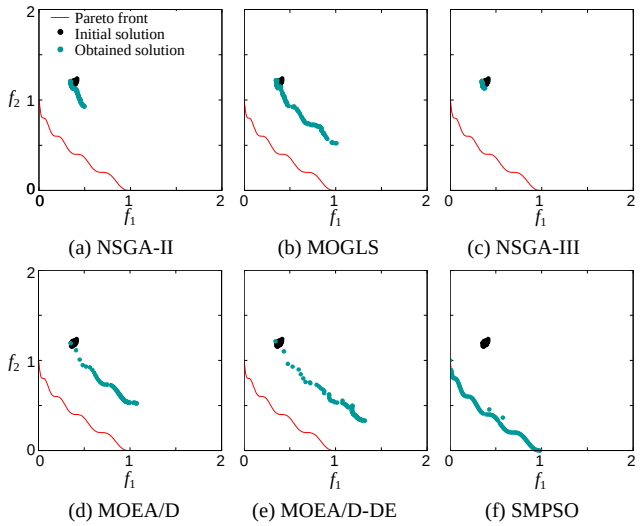
Problem	Modality	Bias	Separability	Scaling	Geometry
NSGA-II$_{Min}$	Deceptive	Polynomial	Separable	Double	Convex-disc
MOGLS$_{Min}$	Multi-modal	--	Non-separable	Same	Concave-disc
NSGA-III$_{Min}$	Uni-modal	Polynomial	Non-separable	Double	Concave
MOEA/D$_{Min}$	Multi-modal	DistanceToPosition	Separable	Exponential	Concave-mixed
MOEA/D-DE$_{Min}$	Uni-modal	PositionToDistance	Separable	Double	Concave-mixed
SMPSO$_{Min}$	Multi-modal	PositionToDistance	Separable	Same	Concave

TABLE VII. OBTAINED HYPERVOLUME ON GENERATED TWO-OBJECTIVE PROBLEMS. THE WORST RESULTS ON EACH PROBLEM ARE SHOWN IN BOLD.

Problem	NSGA-II	MOGLS	NSGA-III	MOEA/D	MOEA/D-DE	SMPSO
NSGA-II$_{Min}$	**7.77**	8.50	7.80	8.29	8.25	8.26
MOGLS$_{Min}$	7.96	**7.49**	7.99	7.60	7.98	8.25
NSGA-III$_{Min}$	414.14	701.28	**217.64**	595.86	701.75	742.94
MOEA/D$_{Min}$	219.23	219.17	219.22	**218.69**	629.55	661.65
MOEA/D-DE$_{Min}$	314.40	307.90	315.17	302.03	**301.71**	315.49
SMPSO$_{Min}$	3.76	3.72	3.74	3.71	3.71	**3.65**

TABLE VIII. OBTAINED HYPERVOLUME ON GENERATED SIX-OBJECTIVE PROBLEMS. THE WORST RESULTS ON EACH PROBLEM ARE SHOWN IN BOLD.

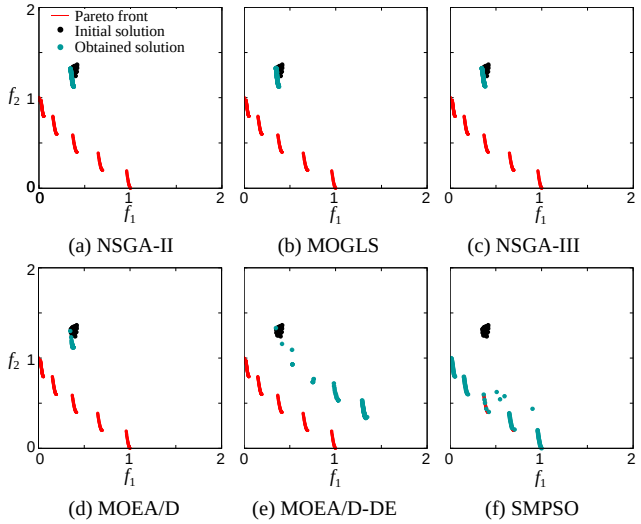
Problem	NSGA-II	MOGLS	NSGA-III	MOEA/D	MOEA/D-DE	SMPSO
NSGA-II$_{Min}$	**4.88E+04**	1.19E+05	1.04E+05	1.15E+05	1.29E+05	1.35E+05
MOGLS$_{Min}$	54.05	**49.19**	52.45	49.45	54.31	52.55
NSGA-III$_{Min}$	1.89E+04	2.59E+04	**1.07E+04**	2.86E+04	8.07E+04	1.09E+05
MOEA/D$_{Min}$	6.01E+20	6.15E+20	8.02E+20	**2.37E+20**	5.16E+20	6.92E+20
MOEA/D-DE$_{Min}$	8.14E+04	9.66E+04	1.11E+05	6.34E+04	**5.04E+04**	7.78E+04
SMPSO$_{Min}$	57.57	59.68	59.47	59.91	58.41	**51.91**



(a) NSGA-II (b) MOGLS (c) NSGA-III

(d) MOEA/D (e) MOEA/D-DE (f) SMPSO

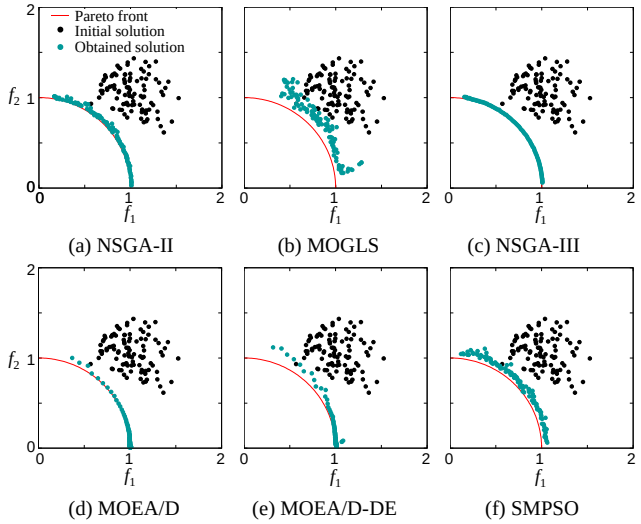
Fig. 3. Obtained results on the two-objective NSGA-III$_{\text{Min}}$.



(a) NSGA-II (b) MOGLS (c) NSGA-III

(d) MOEA/D (e) MOEA/D-DE (f) SMPSO

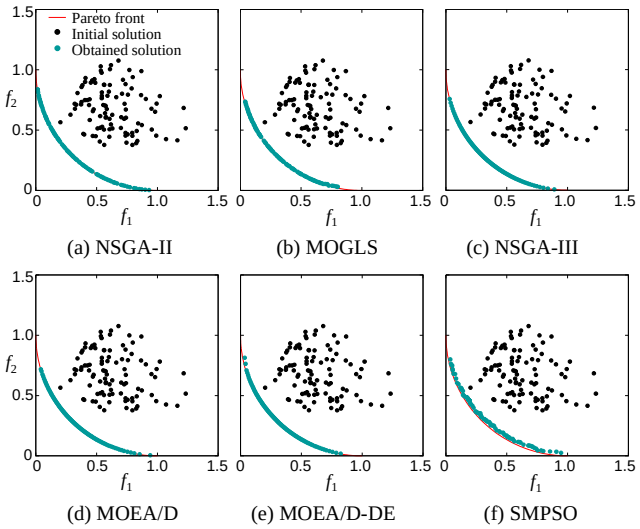
Fig. 4. Obtained results on the two-objective MOEA/D$_{\text{Min}}$.



(a) NSGA-II (b) MOGLS (c) NSGA-III

(d) MOEA/D (e) MOEA/D-DE (f) SMPSO

Fig. 5. Obtained results on the two-objective MOEA/D-DE$_{\text{Min}}$.



(a) NSGA-II (b) MOGLS (c) NSGA-III

(d) MOEA/D (e) MOEA/D-DE (f) SMPSO

Fig. 6. Obtained results on the two-objective SMPSO$_{\text{Min}}$.

C. *Discussion about the Functions Determining the Features of the WFG Problem*

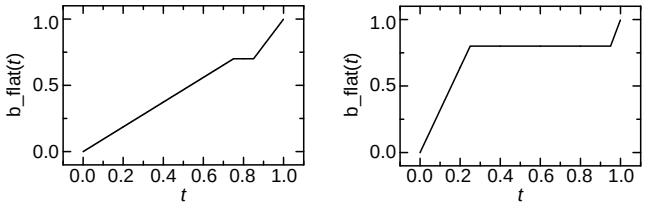
In this paper, the functions determining the features of the WFG problem are utilized for the test problem generation. In order to avoid complex combinations, these functions are picked from WFG1-WFG8. For example, the parameter setting of the polynomial bias function is used the same setting as in WFG1 as shown in Table I. However some of these parameters settings have large relationship with the characteristics of test problems. For discussing about the effect of the parameter setting of the functions, we examine the three different parameter settings of the flat function. The flat function can be specified as follows:

$$\text{b\_flat}(t, A, B, C) = A + \min(0, \lfloor t - B \rfloor)\frac{A(B-t)}{B} - \min(0, \lfloor C - t \rfloor)\frac{(1-A)(t-C)}{1-C}, \quad (5)$$

where t is the input decision variables and A, B, C are control parameters. Values of t between B and C are mapped to A. That is, the values of B and C define the width of the flat region of decision variables. We compare following three settings for the flat function in two-objective NSGA-II$_{\text{Min}}$:

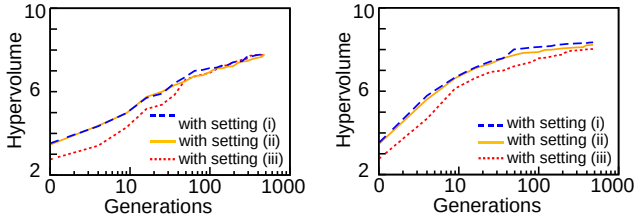
(i) $A = 0.8$, $B = 0.8$, $C = 0.8$ (no flat region),
(ii) $A = 0.8$, $B = 0.75$, $C = 0.85$ (the same setting in WFG1),
(iii) $A = 0.8$, $B = 0.25$, $C = 0.95$.

We show the fitness landscapes of the flat function with the settings (ii) and (iii) in Fig. 7. The obtained results for the three settings are summarized in Fig. 8. From Fig. 7, we can observe that problems have a larger flat region in the setting (iii) compare with the setting (ii). Because of a larger flat region, two-objective NSGA-II$_{\text{Min}}$ with the setting (iii) can be considered as a more difficult problem than one with the setting (ii). In Fig. 8, we can observe the clear performance deterioration of both algorithms in the setting (iii). This observation suggests the importance of the appropriate parameter setting of the functions of WFG toolkit.



(a) Setting (ii) (b) Setting (iii)

Fig. 7. The fitness landscapes of the flat function with two settings.



(a) NSGA-II (b) SMPSO

Fig. 8. Obtained results on the three different settings of the two-objective NSGA-II$_{Min}$.

VI. Conclusions

In this paper, we examined the effectiveness of the automatic test problem generation based on meta-optimization. From the computational experiments, we observed that our meta-optimization method successfully obtained test problems which have the disadvantageous characteristics for their target algorithms. For the generation of test problems, we combined functions determining the features of the WFG problem. Although the parameter settings of each function are picked from WFG1-WFG8, the obtained test problems from meta-optimization have totally different characteristics from WFG1-WFG8. These test problems can be utilized for the touchstone against a development of a more robust algorithm in the future.

In our meta-optimization approach for generating test problems, there is high flexibility in the definition of a fitness calculation and how to create candidate test problems. For the fitness calculation, hypervolume is used as the quality indicator in this study. Other quality indicators such as IGD are also utilized for the fitness evaluation. The influence of the fitness evaluation in meta-optimization is an interesting future research topic. It is also an interesting future research topic that each objective function is designed by the other techniques such as a genetic programming. As discussed in Section V-C, the characteristics of obtained test problems from meta-optimization depend on the scope of problems which can be generated. One important issue we did not discuss in this paper is the relationship between the generated test problems and actual problems faced in practice. It is also an interesting future research topic that generating a simple test problem that has a same feature of the actual problem.

References

[1] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, 2002.

[2] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Trans. on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, vol. 28, no. 3, pp. 392-403, 1998.

[3] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. on Evolutionary Computation*, vol. 18, no. 4, pp. 577-601, 2013.

[4] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. on Evolutionary Computation*, vol. 11, no. 6, pp. 712-731, 2007.

[5] H. Li, and Q. Zhang. "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Trans. on Evolutionary Computation*, vol. 13, no. 2, pp. 284-302, 2009.

[6] A. J. Nebro, J. J. Durillo, J. García-Nieto, C. A. C. Coello, F. Luna, and E. Alba, "SMPSO: A new pso-based metaheuristic for multi-objective optimization," *Computational Intelligence in Miulti-Criteria Decision-Making*, pp. 66-73, 2009.

[7] C. A. C. Coello and G. B. Lamont, *Applications of multi-objective evolutionary algorithms*, vol. 1, World Scientic, 2004.

[8] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," *In Proc. of 2002 IEEE Congress on Evolutionary Computation (World Congress on Computational Intelligence)*, pp. 825-830, 2002.

[9] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari,"Multiobjective optimization test instances for the CEC 2009 special session and competition," Univ. Essex and Nanyang Technol. Univ., Essex, U.K./Singapore, Tech. Rep. CES-487, 2008.

[10] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," *Lecture Notes in Computer Science: Learning and Intelligent Optimization*, vol. 6683, pp. 507-523, Springer, Berlin, 2011.

[11] F. Hutter, T. Stuetzle, K. Leyton-Brown, and H. H. Hoos, "ParamILS: An automatic algorithm configuration framework," *Journal of Artificial Intelligence Research*, vol. 36, pp. 267-306, 2009.

[12] P. Balaprakash, M. Birattari, and T. Stützle, "Improvement strategies for the F-race algorithm: Sampling design and iterative renement," *Lecture Notes in Computer Science*, vol. 4771, pp. 108-122, 2007

[13] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu, "Hyper-heuristics: A survey of the state of the art," *Journal of the Operational Research Society*, vol. 64, no. 12, pp. 1695-1724, 2013.

[14] S. Huband, L. Barone, L. While, and P. Hington, "A scalable multi-objective test problem toolkit," *Lecture Notes in Computer Science: Evolutionary Multi-Criterion Optimization*, vol. 3410, pp. 280-295, Guanajuato, 2005.

[15] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms – A comparative case study," *Lecture Notes in Computer Science: Parallel Problem Solving from Nature*, vol. 1498, pp. 292-301, Springer, Berlin, 1998.

[16] O. Schuetze, X. Esquivel, A. Lara, and C. A. C. Coello, "Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization," *IEEE Trans. on Evolutionary Computation*, vol. 16, no. 4, pp. 504-522, 2012.

[17] O. Schuetze, X. Esquivel, A. Lara, and C. A. C. Coello, "Automatic component-wise design of multi-objective evolutionary algorithms," *IEEE Trans. on Evolutionary Computation* (Available from IEEE Xplore as an Early Access Paper).

[18] Y. Tanigaki, H. Masuda, Y. Setoguchi, Y. Nojima, and H. Ishibuchi, "Algorithm structure optimization by choosing operators in multiobjective genetic local search," *In Proc. of 2015 IEEE Congress on Evolutionary Computation*, pp. 854-861, 2015.

[19] K. Deb, and A. Kumar, "Real-coded genetic algorithms with simulated binary crossover: studies on multimodal and multiobjective problems", *Complex Systems*, vol. 9, no. 6, pp. 431-454, 1995.

[20] M. Hamdan, "On the disruption-level of polynomial mutation for evolutionary multi-objective optimization algorithms", *Computing and Informatics*, vol. 29, pp. 783-800, 2010.