Effect of Solution Information Sharing between Tasks on the Search Ability of Evolutionary Multiobjective Multitasking Algorithms

Ryuichi Hashimoto[1], Naoki Masuyama[1], Yusuke Nojima[1], and Hisao Ishibuchi[2]
[1]Graduate School of Engineering, Osaka Prefecture University, Sakai, Osaka, Japan
{ryuichi.hashimoto@ci., masuyama@, nojima@}cs.osakafu-u.ac.jp
[2]Shenzhen Key Laboratory of Computational Intelligence,
University Key Laboratory of Evolving Intelligent Systems of Guangdong Province,
Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China
hisao@sustech.edu.cn

***Abstract*—A multitask multiobjective optimization problem (MTMOP) has multiple multiobjective problems to be solved simultaneously (i.e., multitasking of multiple multiobjective problems). One approach to such an MTMOP is evolutionary multiobjective multitasking (EMOMT). EMOMT algorithms solve multiple tasks simultaneously in a cooperative manner. They are evolutionary algorithms with multiple sub-populations. Each sub-population corresponds to a single task. During the evolutionary search, the information on each solution is shared with other sub-populations. Some EMOMT algorithms have been developed by focusing on solution information sharing. However, the effect of sharing the solution information on the search ability of EMOMT algorithms is not well examined yet. Through the examination of this effect, it is expected that better EMOMT algorithms than existing ones can be developed. In our previous study, we proposed an island model-based evolutionary single-objective multitasking algorithm. Using our island model, we can analyze the effect of solution information sharing. In this paper, as an extension of our previous study, we develop an island model-based EMOMT algorithm framework for MTMOPs. Through computational experiments under various parameter settings, we examine the effect of solution information sharing on the search ability of our island model.**

***Keywords*—Multitasking; multitask multiobjective optimization; evolutionary multiobjective optimization; island model; information sharing**

I. Introduction

In real-world applications, optimization problems often have multiple conflicting objectives. These problems are called multiobjective optimization problems (MOPs). In general, there is a trade-off relationship between the objectives. As a result, an MOP has no single optimal solution which optimizes all of its objectives simultaneously. Instead, it has a number of Pareto optimal solutions along the trade-off curve. The set of all Pareto optimal solutions in the objective space is called the Pareto front. When our goal is to search for a number of well-distributed Pareto optimal solutions over the entire Pareto front, it is inefficient to search for each Pareto optimal solution by an independent run of an optimization algorithm. Various evolutionary multiobjective optimization algorithms (EMOAs) have been proposed to search for a number of well-distributed Pareto optimal solutions by their single run [1]–[4].

Multitask multiobjective optimization (MTMOO) has recently attracted a lot of attention as a new research direction [5]–[9]. MTMOO algorithms solve multitask multiobjective optimization problems (MTMOPs) with multiple MOPs to be solved simultaneously. Each MOP is called a task. MTMOO algorithms try to solve multiple MOPs (i.e., multiple tasks) by their single run. That is, they try to search for multiple solution sets, each of which is a set of well-distributed Pareto optimal solutions for each MOP. One approach to MTMOPs is evolutionary multiobjective multitasking (EMOMT). Previous studies showed that EMOMT algorithms have higher search ability than single-tasking algorithms [5]–[9].

Some EMOMT algorithms have been developed by focusing on sharing the solution information (i.e., values of decision variables in each solution). In [7], three crossover probabilities are used in inter-task crossover. In [8], a denoising autoencoder is used to transfer a solution in the decision space of one task to that of another task. These algorithms showed promising search ability. Whereas the solution information sharing is a key feature of EMOMT, its effect on the search ability of EMOMT algorithms has not been examined thoroughly in the literature. Thus, it is difficult to clearly explain why multitasking for most tasks outperforms independent runs of single-tasking algorithms on each task. By analyzing the effect of the solution information sharing, we may be able to explain the high search ability of EMOMT algorithms, which may suggest future research directions for further improving the search ability of EMOMT algorithms.

This work was supported by National Natural Science Foundation of China (Grant No. 61876075), the Program for Guangdong Introducing Innovative and Enterpreneurial Teams (Grant No. 2017ZT07X386), Shenzhen Peacock Plan (Grant No. KQTD2016112514355531), the Science and Technology Innovation Committee Foundation of Shenzhen (Grant No. ZDSYS201703031748284), and the Program for University Key Laboratory of Guangdong Province (Grant No. 2017KSYS008).

Corresponding Author: Hisao Ishibuchi (hisao@sustech.edu.cn)

In our previous study [10], we developed an island model-based evolutionary multitasking algorithm for multitask single-objective optimization problems (MTSOPs). We examined the effect of solution information sharing on its search ability for MTSOPs in detail. By using our island model, it is easy to analyze the effect of sharing the solution information among different tasks. In this paper, as an extension of our previous study [10], we propose an island model for solving MTMOPs and examine the effect of solution information sharing on the search ability of the proposed island model for MTMOPs.

This paper is organized as follows. In Section II, we discuss the sharing of solution information over multiple tasks in EMOMT algorithms. In Section III, we propose the island model-based EMOMT algorithm. In Section IV, we examine the effect of solution information sharing over multiple tasks on the search ability of our island model-based EMOMT algorithm through computational experiments. Finally, we conclude this paper in Section V.

II. Multitask Multiobjective Optimization

A. Multiobjective Optimization

Without loss of generality, we assume that each objective is to be minimized. An MOP can be written as follows:

$$\text{Minimize } \boldsymbol{f}(\boldsymbol{x}) = \left(f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ ..., f_m(\boldsymbol{x})\right), \boldsymbol{x} \in \boldsymbol{X}, \quad (1)$$

where $f_i(\boldsymbol{x})$ is the ith objective, and m is the number of objectives. In (1), $\boldsymbol{x}$ and $\boldsymbol{X}$ are a decision variable vector and a decision space, respectively. A number of EMOAs have been developed to efficiently solve MOPs [1]–[4]. Utilizing their population-based search mechanisms, EMOAs try to search for a set of well-distributed solutions over the entire Pareto front of an MOP by their single run.

B. Multitask Multiobjective Optimization

By further utilizing population-based search mechanisms, EMOAs can solve multiple MOPs (i.e., tasks) simultaneously. When these tasks have similar characteristics, it is expected that convergence ability toward the Pareto front for each task is improved by sharing the information about solutions of other tasks. Based on this idea, recently an MTMOO algorithm has been developed to solve multiple tasks simultaneously in a cooperative manner [5].

An MTMOP with k tasks is called a k-task MTMOP. Each task has a number of objectives to be minimized. A k-task MTMOP can be written as follows:

$$\begin{aligned} &\text{Minimize } \{\boldsymbol{f}_1(\boldsymbol{x}_1), \boldsymbol{f}_2(\boldsymbol{x}_2), \ ..., \boldsymbol{f}_k(\boldsymbol{x}_k)\}, \\ &\text{where } \boldsymbol{f}_i(\boldsymbol{x}_i) = \left(f_{i1}(\boldsymbol{x}_i), f_{i2}(\boldsymbol{x}_i), \ ..., f_{im_i}(\boldsymbol{x}_i)\right), \quad (2) \\ &\boldsymbol{x}_i \in \boldsymbol{X}_i \ (i = 1, \ 2, ..., k). \end{aligned}$$

$\boldsymbol{x}_i$ and $\boldsymbol{X}_i$ are a decision variable vector and a decision space of the ith MOP (i.e., ith task), respectively. $\boldsymbol{f}_i$ is an objective function vector of the ith MOP, $f_{ij}(\boldsymbol{x}_i)$ is the jth objective of the ith MOP, and m_i is the number of objectives of the ith MOP.

C. Evolutionary Multiobjective Multitasking Algorithms

One of the main approaches to solve MTMOPs is EMOMT. EMOMT algorithms are EMOAs which solve multiple MOPs in a cooperative manner [5]–[8]. EMOMT algorithms search for a set of well-distributed solutions over the entire Pareto front of each MOP. In most EMOMT algorithms, each solution in a population is assigned to a single MOP (i.e., a single task). Thus, a population can be implicitly divided into multiple sub-populations by the task assignment [10]. Whereas each sub-population is optimized for the corresponding task, the information of solutions in each sub-population is shared with other sub-populations. The solution information sharing among the sub-populations accelerates the convergence of solutions in each sub-population towards the Pareto front of the corresponding task. Therefore, it can be regarded that the sharing of solution information among multiple tasks is the key point for EMOMT algorithms.

Some approaches to solution information sharing have been proposed [5]–[8]. In most of EMOMT algorithms, the information of solution for each task is implicitly shared by applying crossover between solutions for different tasks. A pair of solutions are selected from a set of all solutions, each of which is for a single task. Offspring solutions are generated by applying crossover and mutation to the selected parents. The generated offspring solutions are chosen for some tasks. In this manner, genetic information of a solution for a task can be propagated to solutions for other tasks.

Multiobjective multifactorial evolutionary algorithm (MO-MFEA) is the representative for EMOMT algorithms. In MO-MFEA, nondominated sorting genetic algorithm-II (NSGA-II) [1] can be viewed as being applied to each task. It has been reported in some studies [5], [9] that MO-MFEA showed better performance than NSGA-II. However, it is difficult to examine the effects of solution information sharing because the inter-task migration of solutions in MO-MFEA is not explicitly performed.

III. Island Model for MTMOPs

In [10], we introduced the island model-based evolutionary single-objective multitasking algorithm which can analyze solution information sharing between tasks. In this paper, we propose the island model by introducing a new migration operator to deal with MTMOPs.

A. Migration in the Island Model for MTSOPs

First, we explain our island model-based evolutionary single-objective multitasking algorithm [10]. For the simplicity of explanations, we assume that we have two tasks. The island model is a multi-population based algorithm. The number of sub-populations is the same as the number of tasks to be solved. Each sub-population (i.e., an island) solves the assigned task by using a genetic algorithm. During solving a task in each island, the following migration between islands is periodically performed: A pre-specified number of solutions are randomly selected and copied from each island as migrated solutions. The same number of the worst solutions in terms of the fitness value are removed from each island. Then, the migrated solutions are inserted to the other island.

The goal of single-objective evolutionary algorithms is to search for the single best solution with the best fitness value. Thus, it is a reasonable approach to remove the solutions with the worst fitness values from each island. In contrast, the goal of EMOAs is to search for a set of well-distributed non-dominated solutions over the entire Pareto front. Thus, in the case that the number of removing solutions is larger than the number of dominated solutions, there is a possibility that non-dominated solutions are removed by the migration operators. As a result, the quality of the current solutions to approximate the Pareto front becomes worse. To address this problem, we modify the migration operator for MTMOPs in our island model.

B. Proposed Migration Operator for MTMOPs

Fig. 1 shows the proposed migration operators for MTMOPs. Let us assume that the number of tasks is two, the size of each sub-population is N, the number of migrated solutions is n, and the migration interval is t (i.e., the migration is performed at every t generations). First, a parent pool is generated in each island. Next, the best $(N - n)$ solutions, which are selected based on the solution evaluation criteria in NSGA-II (i.e., non-dominated ranking and crowding distance), are copied from each island. They are added to the parent pool of its own island as shown by (I) in Fig. 1. Next, n solutions are randomly selected as migrated solutions from each island, sent to the other island, and added to the parent pool as shown by (II) in Fig. 1. Then, an offspring population is generated from the parent pool as shown by (III). That is, the offspring population is generated from the migrated n solutions and the best $(N - n)$ solutions in the current population. Finally, the best N solutions are selected from the current population and the offspring population in each island as shown by (IV). Since the worst solutions in the current population can be selected as members of the next population, the deterioration in the approximation quality can be prevented.

As we have already mentioned in this section, our island model has two migration parameters: the migration interval t and the migration size n (i.e., n solutions are migrated from one island to the other island). If the migration is not performed, our island model is the same as the independent execution of the EMOA on each task. The migration is the simplest form of the solution information sharing. Thus, we can easily analyze the effect of solution information sharing by comparing the search ability between our island model and the EMOA.

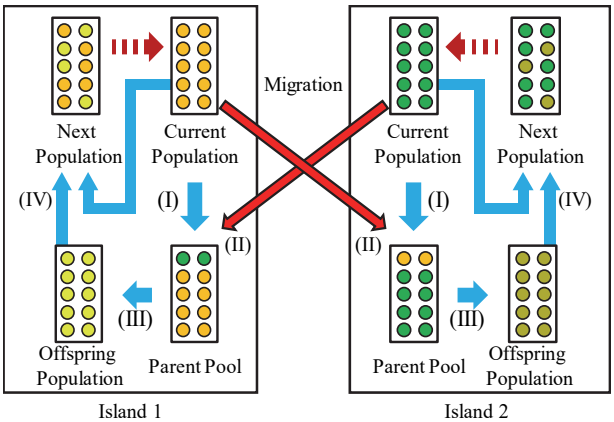


Fig. 1. Migration between islands ($N = 10$, $n = 2$).

IV. Effect of Solution Information Sharing

A. Experimental Settings

In this section, we examine the effect of the solution information sharing among tasks. We apply our island model and NSGA-II to the nine benchmark problems in [9]. Since NSGA-II is a well-known and frequently used representative EMOA and many EMOMT algorithms are based on NSGA-II, we use NSGA-II in our experiments. In our island model, each sub-population is optimized by NSGA-II. Each benchmark problem has a pair of two- or three-objective optimization tasks. Each task has a different distance function in the formulation of the corresponding MOP. Characteristics of each problem (with two tasks) are summarized in Table I. The nine problems can be divided into three categories based on the intersection degree of the global optima of the distance functions between the two tasks: Complete Intersection (CI), Partial Intersection (PI) and No Intersection (NI). CI, PI, and NI mean that the optimal values of decision variables for the distance functions are the same, partially same, and completely different between the two tasks, respectively. The nine problems can be also divided into three categories by the similarity in the fitness landscape of the distance functions between the two tasks: High Similarity (HS), Medium Similarity (MS), and Low Similarity (LS). In the first (CI+HS) problem in Table 1, a good solution for one task is very likely to be a good solution for the other task. However, for the last (NI+LS) problem, it is likely that good solutions for each task are totally different.

Table II shows the experimental settings used in this paper. These experimental settings are the same as [9]. To analyze the effect of the solution information sharing in detail, we use various specifications of the two migration parameters: the migration interval (t) and the migration size (n). We use the inverted generational distance (IGD) [11] as a performance measure. IGD for both the convergence and the spread of a solution set. After computational experiments, we examine which algorithm is significantly better or worse than the other. The differences between the results are assessed by Welch's t-test or Brunner-Munzel test at the 0.05 significance level. Welch's t-test is utilized if the results follow a normal distribution, otherwise Brunner-Munzel test is used.

TABLE I. The Characteristics of Benchmark Problems.

Problem	Intersection degree of global optima between two tasks	The similarity in the fitness landscape of two tasks
CI+HS	Complete Intersection (CI)	High Similarity (HS)
CI+MS	Complete Intersection (CI)	Medium Similarity (MS)
CI+LS	Complete Intersection (CI)	Low Similarity (LS)
PI+HS	Partial Intersection (PI)	High Similarity (HS)
PI+MS	Partial Intersection (PI)	Medium Similarity (MS)
PI+LS	Partial Intersection (PI)	Low Similarity (LS)
NI+HS	No Intersection (NI)	High Similarity (HS)
NI+MS	No Intersection (NI)	Medium Similarity (MS)
NI+LS	No Intersection (NI)	Low Similarity (LS)

TABLE II. EXPERIMENTAL SETTINGS.

Crossover	SBX [12] ($p_c = 0.9$, $\mu_c = 20$)
Mutation	Polynomial mutation [13] ($p_m = 1/d$, $\mu_m = 20$) d is the number of decision variables
Population size	100 for each task
The number of runs	31
Stopping condition	200,000 solution evaluations
Migration interval (t)	{1, 5, 10, 20, 25, 50, 100, 200, 250, 500}
Migration size (n)	{2, 10, 20, 30, 40, 50, 60, 70, 80, 90}

Experimental results on the task 1 in the first benchmark problem in Table I (i.e., CI+HS) are shown in Fig. 2. In Fig. 2, the results of the statistical test between our island model and NSGA-II are shown by the color of each cell. Red cells mean that the difference in the IGD values between these algorithms is statistically significant and the results by our island model are better than by NSGA-II. Blue cells mean that the difference in the IGD values between these algorithms is statistically significant and the results by NSGA-II are better than those by our island model. A white cell means that the difference in the IGD values between the two algorithms is not statistically significant. In the upper-left cells in Fig. 2, a large amount of solution information is shared very frequently. In contrast, in the lower right cells in Fig. 2, a small amount of solution information is shared less frequently. From this figure, we can see that our island model obtained better results than NSGA-II in all specifications of the two migration parameters except for the combinations of the shortest migration interval (i.e., $t = 1$) and the very large migration sizes (i.e., $n = 80$, 90). When the migration size is 90 and the migration interval is 1, NSGA-II obtained better results than our island model. When the migration size is 80 and the migration interval is 1, there is no statistically significant difference. In this paper, we show our experimental results in the same manner as in Fig. 2.

B. Experimental Results on Complete Intersection Problems

Fig. 3 shows the results on the CI+HS problem (Fig. 3 (a) is the same as Fig. 2). In Fig. 3, our island model obtained better results for both tasks than NSGA-II in many specifications of the two migration parameters. Since the two tasks have the same global optima of the distance functions, near-optimal solutions for the distance function of one task are likely to strongly help the search for the other task. Thus, better results were almost always obtained by our island model than NSGA-II. However, our island model was outperformed by NSGA-II when the migration interval is the shortest and the migration size is large (i.e., blue cells in Fig. 3). This can be explained as follows. In our island model, all migrated solutions are re-evaluated in the migrated island. This evaluation process increases the number of solution evaluations per generation, which decreases the total number of generations since the stopping condition is the total number of solution evaluations for a fair comparison. Thus, the performance of our island model is worse than NSGA-II.

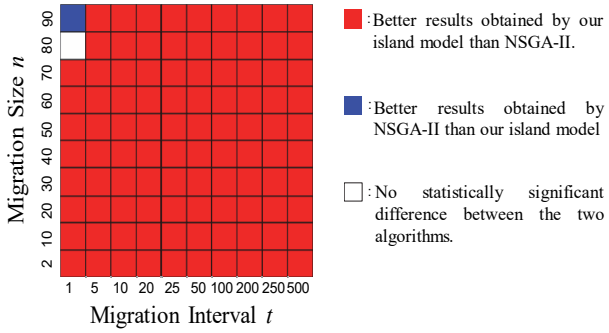
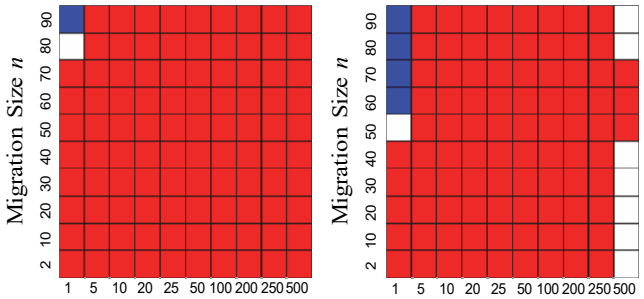


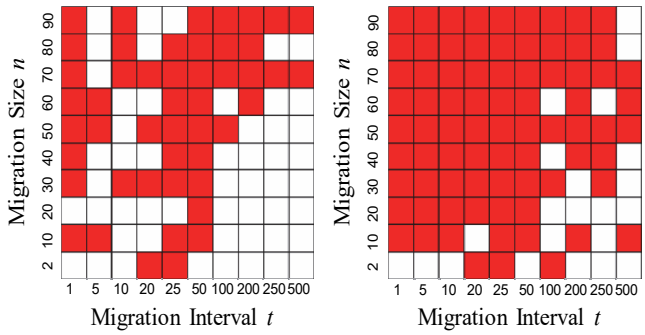
Fig. 2. The results of the statistical test on the task 1 in the first benchmark problem (CI+HS).



(a) Task 1 (b) Task 2

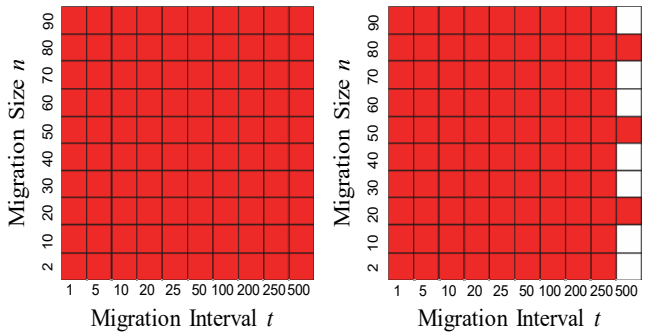
Fig. 3. Experimental results on the CI+HS problem

Fig. 4 and Fig. 5 show the results on the CI+MS and CI+LS problems, respectively. Similar to the CI+HS problem in Fig. 3, better results were obtained by our island model in many specifications of the two migration parameters in Fig. 4 and Fig. 5. One difference between those figures and Fig. 3 is that better results were obtained by our island model in Fig. 4 and Fig. 5 even when the migration interval is one and the migration size is very large. Explanations for this observation



(a) Task 1 (b) Task 2

Fig. 4. Experimental results on the CI+MS problem.



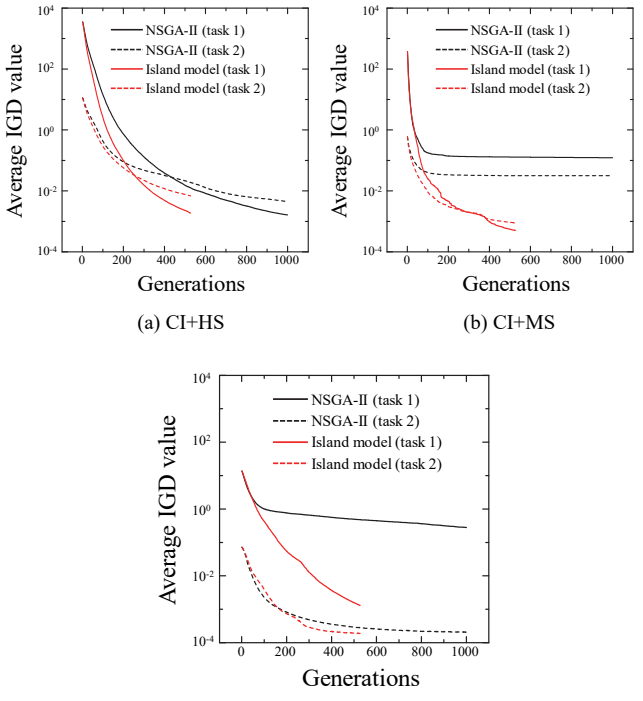
(a) Task 1 (b) Task 2

Fig. 5. Experimental results on the CI+LS problem.

are difficult. One possible reason is the diversity improvement by the frequent migration of many solutions. If NSGA-II converges to a local Pareto front in the early generations, the search ability of NSGA-II is enhanced by the diversity improvement (i.e., migration). The decrease in the number of generations by the migration is not a big problem if NSGA-II converges to a local Pareto front in the early generations.

To confirm the above hypothesis, in Fig. 6, we show the average IGD values at each generation over 31 runs of each algorithm on each of the three CI problems. In Fig. 6, our island model uses the shortest migration interval (i.e., t = 1) and the largest migration size (i.e., n = 90). In Fig. 6 (a), the average IGD value by NSGA-II for each task continues to decrease over the 1,000 generations. In contrast, in Fig. 6 (b), the decrease in the average IGD value by NSGA-II for each task clearly slows down around the 100th generation. It is likely that many runs of NSGA-II converged to a local Pareto front (or a part of the true Pareto front) around the 100th generation. As a result, our island model with about 500 generations outperforms NSGA-II with 1,000 generations in Fig. 6 (b). We have similar observations in Fig. 6 (c). The experimental results in Fig. 6 suggest that the diversity improvement is one advantage of our island model.

Figs. 3-5 also show that better results were obtained by our island model regardless of the similarity in the fitness landscape of the distance function between the two tasks (i.e., HS, MS, or LS). This observation suggests that the similarity in the fitness landscape does not have a large impact on the search performance of our island model when the distance function of each task has the same optimal solutions (i.e., CI).



(a) CI+HS (b) CI+MS

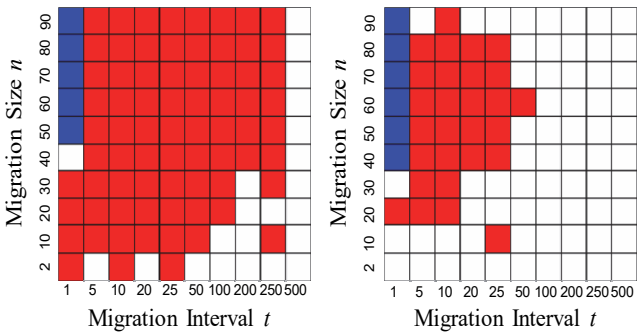
(c) CI+LS

Fig. 6. Average IGD values at each generation of our island model (t = 1, n = 90) and NSGA-II on the three CI problems.

C. *Experimental Results on Partial Intersection Problems*

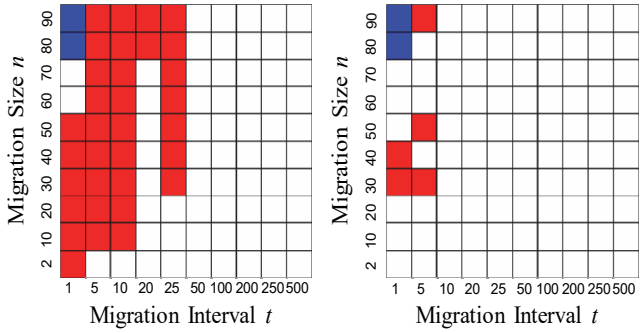
Experimental results on the three PI problems are shown in Figs. 7-9. The performance of our island model was equal to or better than that of NSGA-II in many specifications of the two migration parameters except for the case when the migration size is large and the migration frequency is high. Since some decision variables have the same optimal values between the two tasks, similar results to Figs. 3-5 on the CI problems were obtained in Figs. 7-9 on the PI problems. The main difference of the results between the PI and CI problems is that there exist a lot of cells with no statistically significant difference in Figs. 7-9. Since some decision variables have different optimal values between the two tasks, the positive effect of the migration on the search ability of our island model is not so large in Figs. 7-9 if compared with the case of the same optimal values in Figs. 3-5. As a result, Figs. 7-9 have a large number of cells with no statistically significant difference between NSGA-II and our island model with low migration frequency. This observation suggests that our island model needs to perform the migration frequently (but not very frequently) for the PI problems, especially, in Fig. 7 (b), Fig. 8 (a) and Fig. 8 (b).

Unlike Fig. 7 and Fig. 8, Fig. 9 (a) shows that there exist a large number of cells with no statistically significant difference when the migration interval is short. In addition, Fig. 9 (b) shows that our island model always obtained better results than NSGA-II for the task 2. This may be explained by the following reason: the degree of the convergence to the optimal solution of the distance function is totally different between the tasks. Fig. 10 shows all the solutions in the final generation obtained by 31 runs for each task in the PI+LS problem. Since



(a) Task 1 (b) Task 2

Fig. 7. Experimental results on the PI+HS problem.



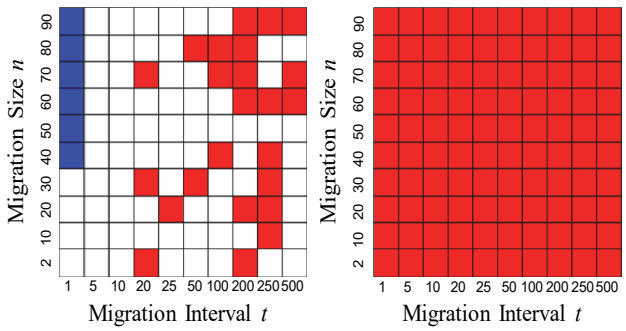
(a) Task 1 (b) Task 2

Fig. 8. Experimental results on the PI+MS problem.

the nine benchmark problems were designed by carefully choosing a distance function for each task, the distribution of obtained solutions of each task in the distance variable space tends to be different between the algorithms. In addition, there exists no dependence among the distance variables. Thus, we show the solutions in the two randomly selected distance variables space (x_2-x_3 space in this case) in Fig. 10. From Fig. 10, we can see that the population for the task 1 obtained by NSGA-II converged to the optimal solution, and the population for the task 2 obtained by NSGA-II did not converge to the optimal solution even in the final generation (i.e., the task 1 is an easy task and the task 2 is a difficult task). In the case of migration from the task 1 to the task 2, migrating solutions often have better objective function values than the solutions of the task 2. In contrast, in the case of migration from the task 2 to the task 1, almost all the migrating solutions have worse objective function values than the solutions of the task 1. From the above reasons, different from Fig. 7 and Fig. 8, there exist many white cells when the migration interval is short in Fig. 9 (a) (i.e., the task 1 in PI+LS) and all cells are colored by red in Fig. 9 (b) (i.e., the task 2 in PI+LS). These results suggest that if we have a difficult task and an easy task, solution information sharing is likey have a positive impact on solving the difficult task, while it may have a negative impact on solving the easy task in the early generation.

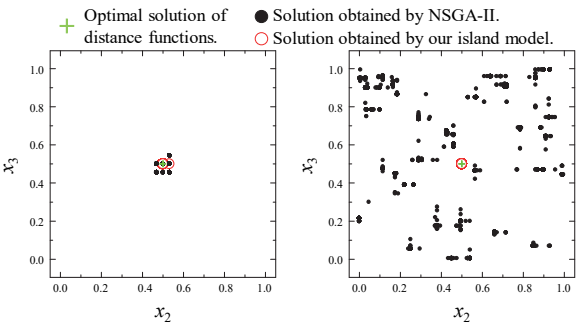
D. Experimental Results on No Intersection Problems

Figs. 11-13 show experimental results on the NI problems. These problems have different global optimal solutions of the distance functions. Intuitively, when the two tasks have different optimal solutions, solutions for one task are useless for the other task. Thus, it is likely that the migration deteriorates the search ability of our island model. However, NSGA-II did not outperform our island model in many parameter specifications except for Fig. 13 (b). Even in Fig. 13 (b), similar results are obtained by the two algorithms in many cases (i.e., there are many white cells). In Figs. 11, 12 (a) and 13 (a), better results were obtained by our island model. These observations can be explained by the distance between the optimal solutions for the distance functions in the two tasks. Whereas the two tasks have different optimal solutions, they are close in the decision space. Fig. 14 shows the optimal solution of the distance function in each task of the NI+HS problem. Whereas the two optimal solutions in Fig. 14 are different, the migration of near-optimal solutions for one task will help the search for the other task. As a result, our island model outperformed NSGA-II. This observation may suggest the necessity of benchmark problems where the distance of the two optimal solutions can be arbitrarily specified.



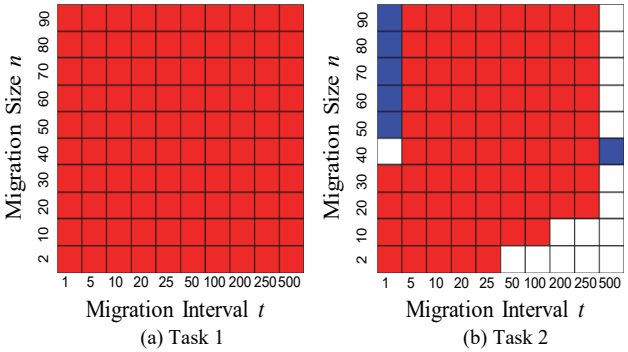
(a) Task 1 (b) Task 2

Fig. 9. Experimental results on the PI+LS problem.



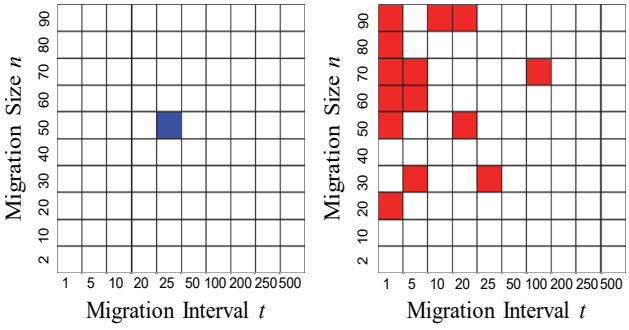
(a) Task 1 (b) Task 2

Fig. 10. All solutions obtained by 31 runs of our island model (t = 5, n = 10) and NSGA-II for each task in the PI+LS problem.



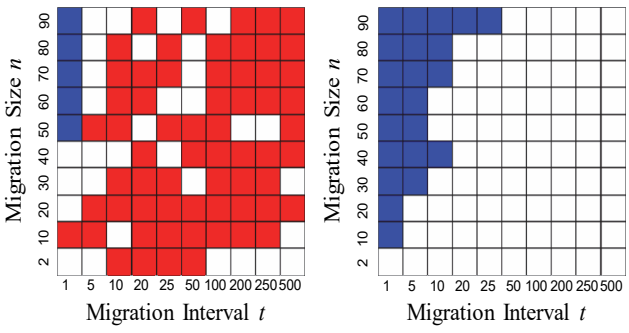
(a) Task 1 (b) Task 2

Fig. 11. Experimental results on the NI+HS problem.



(a) Task 1 (b) Task 2

Fig. 12. Experimental results on the NI+MS problem.



(a) Task 1 (b) Task 2

Fig. 13. Experimental results on the NI+LS problem.

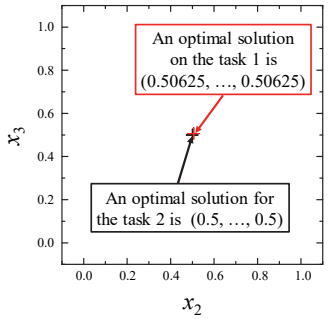


Fig. 14. Optimal solutions of the two tasks in the NI+HS problem.

E. *Discussions on Experimental Results*

Let us discuss overall results obtained by our island model and NSGA-II on the nine benchmark problems. The results on all benchmark problems are summarized in Fig. 15. In Fig. 15, red cells show that better results were obtained by our island model than NSGA-II on at least 9 out of the 18 tasks in the nine benchmark problems. Blue cells show that better results were obtained by NSGA-II than our island model on at least one task. Cells with both blue and red colors mean that the above two results were observed simultaneously (i.e., better results were obtained by our island model on at least nine tasks and by NSGA-II on at least one task).

From Fig. 15, we can see that the search ability of our island model is equal to or better than that of NSGA-II on all the tasks except for some cells around the upper-left corner where many solutions are migrated frequently. Except for these cells with blue color (and with both red and blue colors), the performance of NSGA-II with the migration operator (i.e., our island model) is not deteriorated. In a wide range of parameter specifications, the performance of NSGA-II is improved by the use of our island model for at least a half of the 18 examined tasks in the nine benchmark problems. As we explained in Section III, the difference between our island model and NSGA-II is the use of migration in our island model. Actually, when the migration size is zero or the migration interval is infinity, our island model is the same as NSGA-II. Thus the behavior of our island model with the parameter specifications around the bottom-right corner in Fig. 15 is similar to NSGA-II (since only two solutions are migrated every 500 generations). This is the reason why the cells around the bottom-right corner are white (i.e., no statistically significant difference between the two algorithms).

Except for the above-mentioned extreme parameter settings, our island model with a wide range of parameter specifications outperformed NSGA-II on at least a half of the 18 tasks and was not outperformed on any tasks in Fig. 15. This observation shows that the migration of solutions in our island model has a clear positive impact on the search ability of our island model in many cases even when the two parameters are not fine tuned. No negative effect of the migration is observed in 76 out of the 100 parameter settings in Fig. 15 even for a single task in the 18 tasks. That is, the migration has a large positive effect and almost no negative effect on the performance of our island model over a wide range of parameter settings when it is applied to the nine benchmark problems.

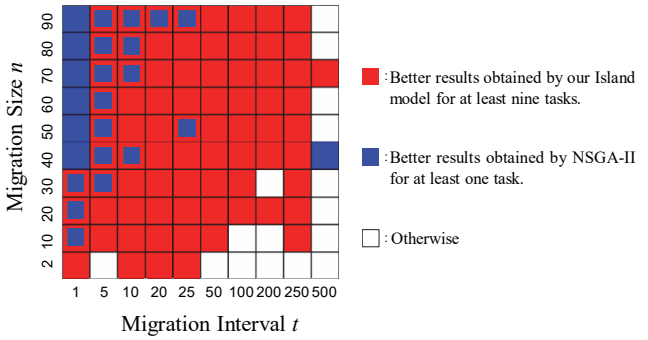


Fig. 15. A summary of the experimental results in Section IV.

V. Conclusion Remarks

In this paper, we first proposed the island model-based EMOMT algorithm. Next, to examine the effect of solution information sharing among multiple tasks, we applied our island model and NSGA-II to the nine benchmark problems in [9]. The main difference between the two algorithms is that the migration is periodically performed between the two tasks in our island model. From the experimental results, we observed the following:

1) When a large number of solutions were migrated to the other task with high frequency, the performance of our island model was deteriorated on many tasks in the benchmark problems (i.e., too much migration is not good in our island model).

2) When we have a difficult task and an easy task to be solved simultaneously, the search ability of our island model on the difficult task can be improved by migrating even a small number of solutions from the easy task to the difficult task. Thus, migration is useful especially when the difficulty of each task is totally different.

3) The migration of solutions from the other task can help the population to escape from a local Pareto front by increasing the diversity of solutions in the population.

4) When some decision variables have the same optimal values between two tasks, the migration in our island model improved its performance (if the migration is not too much or too little).

5) Even if two tasks have different optimal solutions, the search ability of our island model did not deteriorate if the migration is not too much.

These observations were obtained from our island model based on NSGA-II. An interesting future study is to examine the effect of the migration on the performance of our island model implemented with other EMOAs such as MOEA/D [2]. Since each EMOA has its own structure, the framework of our island may need some minor changes. Different observations can be obtained depending on the choice of an EMOA in our island model. Our analysis in this paper also showed some special features of the nine benchmark problems. Some of them are not necessarily appropriate as benchmark problems.

One is the separability of decision variables into distance variables and position variables. Another is the closeness of the optimal solutions in the decision space between two tasks. The creation of more general and/or realistic benchmark problems is an important future research topic.

REFERENCES

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

[2] Q. Zhang and L. Hui, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, 2007.

[3] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *Eur. J. Oper. Res.*, vol. 181, no. 3, pp. 1653–1669, 2007.

[4] 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 Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, 2014.

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

[6] J. Mo, Z. Fan, W. Li, Y. Fang, Y. You, and X. Cai, "Multi-factorial evolutionary algorithm based on M2M decomposition," *Lect. Notes Comput. Sci.*, vol. 10593 LNCS, pp. 134–144, 2017.

[7] C. Yang, J. Ding, K. C. Tan, and Y. Jin, "Two-stage assortative mating for multi-objective multifactorial evolutionary optimization," in *Proc. of Annual Conference on Decision and Control*, 2017, pp. 76–81.

[8] L. Feng, L. Zhou, J. Zhong, A. Gupta, Y. S. Ong, K. C. Tan, and A. K. Qin, "Evolutionary multitasking via explicit autoencoding," *IEEE Trans. Cybern.*, vol. 49, pp. 3457-3470, 2019.

[9] Y. Yuan, Y. S. Ong, L. Feng, A. K. Qin, A. Gupta, B. Da, Q. Zhang, K. C. Chen, Y. Jin, and H. Ishibuchi, "Evolutionary multitasking for multiobjective continuous optimization : Benchmark problems, performance metrics and baseline results," in *Technical Report*.

[10] R. Hashimoto, H. Ishibuchi, N. Masuyama, and Y. Nojima, "Analysis of evolutionary multi-tasking as an island model," in *Proc. of the Genetic and Evolutionary Computation Conference Companion*, 2018, pp. 1894–1897.

[11] S. Jiang, Y. S. Ong, J. Zhang, and L. Feng, "Consistencies and contradictions of performance metrics in multiobjective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2391–2404, 2014.

[12] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, no. 2, pp. 115–148, 1995.

[13] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Comput. Sci. Informatics*, vol. 26, no. 1, pp. 30–45, 1996.