Modification of Evolutionary Multiobjective Optimization Algorithms for Multiobjective Design of Fuzzy Rule-Based Classification Systems

Kaname Narukawa, Yusuke Nojima, and Hisao Ishibuchi
Graduate School of Engineering, Osaka Prefecture University
1-1 Gakuen-cho, Sakai, Osaka 599-8531, Japan
kaname@ci.cs.osakafu-u.ac.jp, {nojima, hisaoi}@cs.osakafu-u.ac.jp

***Abstract*- We examine three methods for improving the ability of evolutionary multiobjective optimization (EMO) algorithms to find a variety of fuzzy rule-based classification systems with different tradeoffs with respect to their accuracy and complexity. The accuracy of each fuzzy rule-based classification system is measured by the number of correctly classified training patterns while its complexity is measured by the number of fuzzy rules and the total number of antecedent conditions. One method for improving the search ability of EMO algorithms is to remove overlapping rule sets in the three-dimensional objective space. Another method is to choose similar rule sets as parents for crossover operations. The other method is to bias the selection probability of parents toward rule sets with high accuracy. The effectiveness of each method is examined through computational experiments on benchmark data sets.**

I. INTRODUCTION

Evolutionary multiobjective optimization (EMO) is one of the most active research areas in the field of evolutionary computation [1], [2]. The main advantage of EMO algorithms over classical multiobjective optimization methods is that a large number of non-dominated solutions can be obtained by a single run of EMO algorithms. This advantage has been utilized to find non-dominated fuzzy rule-based classification systems with different accuracy-complexity tradeoffs in our former studies [3]-[7]. Human users can visually observe the accuracy-complexity tradeoff structure of obtained non-dominated fuzzy rule-based classification systems for a particular pattern classification problem. The multiobjective approach to the design of fuzzy rule-based classification systems can be easily applied to the design of crisp rule-based classification systems [8]. EMO algorithms have also been used to find non-dominated neural networks [9], [10] and non-dominated feature sets [11], [12].

When we try to improve the search ability of EMO algorithms, we have to take into account two goals. One goal is to improve the convergence of solutions to the Pareto front. This goal can be rephrased as driving solutions to the Pareto front as close as possible. The other goal is to increase the diversity of solutions (i.e., to expand the population in the objective space as large as possible). Recently developed EMO algorithms such as SPEA [13] and NSGA-II [14] share some common characteristic features to simultaneously achieve these two goals. One is the use of the Pareto-dominance relation to evaluate each solution. Larger fitness values are assigned to solutions with better (i.e., smaller) ranks using the Pareto-dominance relation because smaller rank solutions can be viewed as being closer to the Pareto front. Diversity-maintenance mechanisms are also used in many EMO algorithms. Larger fitness values are assigned to solutions in less crowded regions in the objective space. Recently the necessity of elitism in EMO algorithms has been widely recognized in the literature (e.g., see Deb [1]). Elitism for multiobjective optimization is to store non-dominated solutions as an archive (i.e., secondary) population separately from the current population or to store them as a part of the current population. Stored non-dominated solutions are used to generate new offspring solutions. It is very difficult to implement non-elite EMO algorithms with high search ability.

The NSGA-II algorithm of Deb et al. [14] seems to be the most well-known and frequently used EMO algorithm in the literature (e.g., see the proceedings of recent conferences on evolutionary computation such as CEC-2004, GECCO-2004, PPSN-2004 and EMO-2005). We also used this algorithm for the multiobjective design of fuzzy and non-fuzzy rule-based classification systems in recent studies [6]-[8] while we used a simple EMO algorithm in earlier studies [3]-[5]. In the NSGA-II algorithm, each solution is evaluated based on the Pareto-dominance relation and a crowding measure. Elitism is also implemented in the NSGA-II algorithm based on the Pareto-dominance relation and the crowding measure. While the NSGA-II algorithm is one of the best EMO algorithms with respect to both the convergence to the Pareto front and the diversity of solutions, it was reported in [15]-[17] that the performance of the NSGA-II algorithm with respect to the diversity of solutions can be further improved by a similarity-based mating scheme for multiobjective knapsack problems.

The goal of this paper is to improve the search ability of the NSGA-II algorithm to find a variety of fuzzy rule-based classification systems. We examine three methods to achieve this goal. One is the removal of overlapping rule sets in the objective space, another is the selection of similar rule sets as parents for crossover, and the other is the selection bias toward rule sets with high accuracy. The effect of each method is examined through computational experiments on benchmark data sets from the UC Irvine Machine Learning Repository where the NSGA-II algorithm is used to find non-dominated fuzzy rule-based classification systems of the three-objective fuzzy rule selection problem in [4]-[7].

II. THREE-OBJECTIVE FUZZY RULE SELECTION

In this section, we briefly describe three-objective fuzzy rule selection to which the NSGA-II algorithm is applied in the next section. Let us assume that we have m labeled patterns $\mathbf{x}_p = (x_{p1}, ..., x_{pn})$, $p = 1, 2, ..., m$ from M classes as training data. We use fuzzy rules of the following type for our n-dimensional pattern classification problem:

$$\text{Rule } R_q: \text{If } x_1 \text{ is } A_{q1} \text{ and ... and } x_n \text{ is } A_{qn} \text{ then Class } C_q \text{ with } CF_q,\ q = 1, 2, ..., N_{\text{rule}}, \quad (1)$$

where R_q is the label of the q-th fuzzy rule, $\mathbf{x} = (x_1, ..., x_n)$ is an n-dimensional pattern vector, A_{qi} is an antecedent fuzzy set, C_q is a consequent class, CF_q is a rule weight, and N_{rule} is the number of fuzzy rules. The consequent class C_q and the rule weight CF_q of each rule R_q are specified from compatible training patterns with its antecedent part $\mathbf{A}_q = (A_{q1}, ..., A_{qn})$ in a heuristic manner [18], [19].

As antecedent fuzzy sets, we use 14 triangular fuzzy sets from four homogeneous fuzzy partitions in Fig. 1. We also use *"don't care"* as an additional antecedent fuzzy set. As a result, each antecedent fuzzy set A_{qi} in (1) assumes one of the 15 alternative fuzzy sets. This means that there exist 15^n possible combinations of the 15 antecedent fuzzy sets for our n-dimensional pattern classification problem.

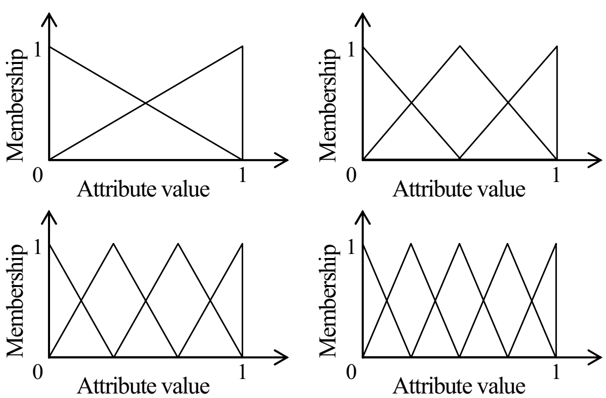

Fig. 1. Homogeneous fuzzy partitions.

Since it is very difficult (at least very time-consuming) to examine all the 15^n combinations of the antecedent fuzzy sets for high-dimensional pattern classification problems, we only examine short fuzzy rules with only a few antecedent conditions. In computational experiments of this paper, we examine fuzzy rules of length 3 or less. Only for the sonar data set with 60 attributes, we examine fuzzy rules of length 2 or less. A prespecified number of promising candidate fuzzy rules are selected from those short fuzzy rules using a heuristic rule evaluation measure. See [19], [20] for various heuristic rule evaluation measures of fuzzy rules. In this paper, we use a heuristic measure used in an iterative fuzzy genetics-based machine learning algorithm called SLAVE [21]. That is, we choose a prespecified number of promising short fuzzy rules for each class using the SLAVE measure. In computational experiments of this paper, 300 fuzzy rules are chosen for each class as candidate rules for multiobjective fuzzy rule selection.

Let us assume that N fuzzy rules have already been chosen as candidate rules (i.e., N/M candidate rules for each class where M is the number of classes). Any subset S of the N candidate rules can be represented by a binary string of length N as $S = s_1 s_2 \cdots s_N$ where $s_j = 1$ and $s_j = 0$ mean that the j-th candidate rule is included in S and excluded from S, respectively.

The three-objective fuzzy rule selection problem in [4]-[7] is written as follows (also see [19]):

$$\text{Maximize } f_1(S) \text{ and minimize } f_2(S) \text{ and } f_3(S), \quad (2)$$

where $f_1(S)$ is the number of correctly classified training patterns by S, $f_2(S)$ is the number of fuzzy rules in S, and $f_3(S)$ is the total number of antecedent conditions of fuzzy rules in S. Since the number of antecedent conditions of each rule is referred to as the rule length, $f_3(S)$ can be viewed as the total rule length. Each training pattern $\mathbf{x}_p$ is classified by a single winner rule R_w chosen from the rule set S as

$$\mu_{\mathbf{A}_w}(\mathbf{x}_p) \cdot CF_w = \max\{\mu_{\mathbf{A}_q}(\mathbf{x}_p) \cdot CF_q \mid R_q \in S\}, \quad (3)$$

where $\mu_{\mathbf{A}_q}(\mathbf{x}_p)$ is the compatibility grade of the training pattern $\mathbf{x}_p$ with the antecedent part $\mathbf{A}_q = (A_{q1}, ..., A_{qn})$ of the fuzzy rule R_q. We use the product operator to calculate the compatibility grade $\mu_{\mathbf{A}_q}(\mathbf{x}_p)$ as

$$\mu_{\mathbf{A}_q}(\mathbf{x}_p) = \mu_{A_{q1}}(x_{p1}) \cdot \ ... \ \cdot \mu_{A_{qn}}(x_{pn}), \quad (4)$$

where $\mu_{A_{qi}}(x_{pi})$ is the compatibility grade of the attribute value x_{pi} with the antecedent fuzzy set A_{qi}.

The NSGA-II algorithm is used to find a large number of non-dominated rule sets of the three-objective fuzzy rule selection problem in (2). As in [4]-[7], we use two problem-specific heuristics to efficiently decrease the number of fuzzy rules in each rule set S during the execution of the NSGA-II algorithm. One is biased mutation probabilities where a larger probability is assigned to the mutation from 1 to 0 than that from 0 to 1. The other is the removal of unnecessary fuzzy rules. Since we use the single winner-based method for classifying each training pattern, some fuzzy rules in S may be chosen as winner rules for no training patterns. We can remove those fuzzy rules without degrading the number of correctly classified training patterns (i.e., $f_1(S)$). At the same time, the removal of such an unnecessary fuzzy rule decreases the number of rules (i.e., $f_2(S)$) and the total rule length (i.e., $f_3(S)$). Thus we remove all fuzzy rules that are not selected as winner rules for any training patterns from the rule set S. The removal of unnecessary fuzzy rules is performed for each rule set S after $f_1(S)$ is calculated and before $f_2(S)$ and $f_3(S)$ are calculated.

III. THE NSGA-II ALGORITHM AND ITS MODIFICATION

In this section, we first describe the NSGA-II algorithm of Deb et al. [14]. Then we explain three methods for improving its search ability to find a variety of non-dominated rule sets of the three-objective fuzzy rule selection problem in (2).

A. The NSGA-II Algorithm

The outline of the NSGA-II algorithm is shown in Fig. 2. First an initial population P is generated in line 1. Pairs of parent solutions are chosen from the current population P in line 3. The set of the selected pairs of parent solutions is denoted by C in line 3. Crossover and mutation operations are applied to each pair in C to generate the offspring population C' in line 4. The next population is constructed by choosing good solutions from the merged population $P \cup C'$. The Pareto-dominance relation and a crowding measure are used to evaluate each solution in the current population P in line 3 and the merged population $P \cup C'$ in line 5. Elitism is implemented in line 5 by choosing good solutions as members in the next population from the merged population $P \cup C'$.

```
1:  P := Initialize (P)
2:  while the stop_criterion is not satisfied do
3:      C := SelectFrom (P)
4:      C' := Vary (C)
5:      P := Replace (P ∪ C')
6:  end while
7:  return (P)
```

Fig. 2. Outline of the NSGA-II algorithm.

The main characteristic feature of the NSGA-II algorithm is its fitness assignment mechanism to each solution. In line 3, pairs of parent solutions are chosen from the current population P by the binary tournament selection scheme where each solution is evaluated in the following manner. The first rank (i.e., rank 1) is assigned to all non-dominated solutions in the current population P. All solutions with the first rank are tentatively removed from the current population P. The second rank (i.e., rank 2) is assigned to all non-dominated solutions in the remaining current population. All solutions with the second rank are tentatively removed from the remaining current population. In this manner, smaller rank values are assigned to better solutions. In the binary tournament selection scheme of the NSGA-II algorithm, the better solution with the smaller rank value is chosen as a parent from two solutions. When the two solutions have the same rank, their crowding measures are calculated using all solutions with the same rank in the current population P. In the NSGA-II algorithm, a crowding measure of each solution is defined as the sum of the distances from adjacent solutions with the same rank. More specifically, two adjacent solutions of each solution are identified with respect to each objective. Then the distance between those adjacent solutions is calculated on each objective and summed up over all objectives for calculating the crowding measure. For each extreme solution with the maximum or minimum value of at least one objective among the same rank solutions, an infinitely large value is assigned to such an extreme solution as its crowding measure because one of the two adjacent solutions cannot be identified. Solutions with larger values of the crowding measure are viewed as being better because those solutions are not located in crowded regions in the objective space.

The Pareto-dominance relation and the crowding measure are also used for the generation update in line 5. That is, the selection of good solutions from the merged population $P \cup C'$ is based on the rank assigned to each solution using the Pareto-dominance relation. The crowding measure is calculated only when solutions with the same rank are to be compared. The next population is constructed in line 5 from the merged population $P \cup C'$ in the following manner. The first rank is assigned to all non-dominated solutions in the merged population. All solutions with the first rank are removed from the merged population and added to the next population. The second rank is assigned to all non-dominated solutions in the remaining merged population. All solutions with the second rank are removed from the remaining merged population and added to the next population. In this manner, better solutions with respect to the Pareto-dominance relation are chosen and added to the next population. If the number of the added solutions to the next population exceeds the population size, solutions with the worst rank in the next population are sorted using the crowding measure calculated for each solution with the same rank. Solutions with the worst rank are removed from the next population in an increasing order of their crowding measures until the number of remaining solutions in the next population becomes the population size. For further descriptions of the NSGA-II algorithm, see Deb [1] and Deb et al. [14].

B. Removal of Overlapping Solutions

In the above-mentioned generation update procedure in line 5 of the NSGA-II algorithm, the first rank is assigned to all non-dominated solutions even when some of them are overlapped with each other in the objective space (i.e., even when some of them have the same objective vector). Many non-dominated solutions with the same objective vector may have a bad effect on the search ability of the NSGA-II algorithm with respect to the diversity of solutions. A direct method for decreasing this bad effect is to construct the next population using different solutions in the objective space. That is, only a single solution can remain in the next population among multiple solutions with the same objective vector. This method can be easily combined into the NSGA-

II algorithm. The same care should be taken in constructing an initial population as well as the next population. That is, we should construct the initial population using different solutions in the objective space. When all solutions in the initial population are different in the objective space, we can always construct the next population with no overlapping solutions from the merged population $P \cup C'$.

C. Recombination of Similar Parents

It was reported in [15] that the recombination of similar parents improved the performance of the NSGA-II algorithm for multiobjective knapsack problems and multiobjective scheduling problems. A similarity-based mating scheme in Fig. 3 was used in [15] to choose similar parents. In this mating scheme, the first parent (i.e., Parent A in Fig. 3) is selected by the binary tournament selection scheme in the same manner as in the NSGA-II algorithm. On the other hand, its mate (i.e., Parent B in Fig. 3) is chosen in the following manner. First β candidates are selected by iterating the binary tournament selection scheme β times. Then the most similar candidate to Parent A is chosen as Parent B from the β candidates. The similarity between Parent A and each candidate is calculated by the Euclidean distance between them in the objective space. The selection bias toward similar parents is adjustable by the value of β in this mating scheme.

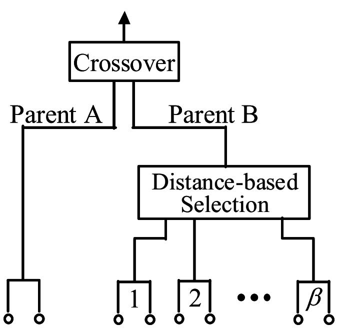

Fig. 3. Mating scheme for choosing similar parents.

D. Selection of Extreme and Similar Parents

The similarity-based mating scheme in Fig. 3 was extended in [16] to choose an extreme solution as Parent A as shown in Fig. 4 where its α candidates are selected by iterating the binary tournament selection scheme α times. In the original extension in [16], the Euclidean distance from each candidate to the average objective vector over the α candidates was calculated in the objective space. The most dissimilar candidate from the average objective vector was chosen as Parent A in [16]. In this paper, we choose the candidate with the highest classification accuracy among the α candidates. This is because the other extreme solution with the lowest complexity is usually an empty rule set with no fuzzy rules (i.e., with a zero classification rate).

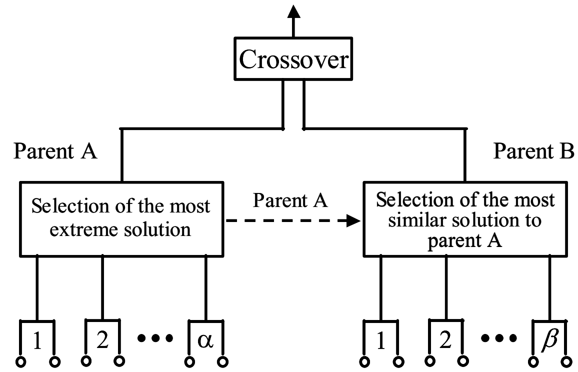

Fig. 4. Mating scheme for choosing extreme and similar parents. Among α candidate, the best candidate with the highest classification accuracy is chosen as Parent A.

IV. COMPUTATIONAL EXPERIMENTS

Through computational experiments on six data sets in Table 1 from the UC Irvine Machine Learning Repository, we examine the effects of the three methods in the previous section on the performance of the NSGA-II algorithm for the three-objective fuzzy rule selection problem. Each method is combined with the NSGA-II algorithm. Thus we have three variants of the NSGA-II algorithm. Since the aim of our computational experiments in this paper is to examine the search ability of the three variants of the NSGA-II algorithm, all patterns in each data set are used as training patterns. As we have already mentioned, 300 fuzzy rules are generated as candidate rules for each class using the SLAVE measure.

TABLE 1. DATA SETS IN COMPUTATIONAL EXPERIMENTS.

Data set	Attributes	Patterns	Classes
Breast W	9	683*	2
Diabetes	8	768	2
Glass	9	214	6
Heart C	13	297*	5
Sonar	60	208	2
Wine	13	178	3

* Incomplete patterns with missing values are not included.

The three variants of the NSGA-II algorithm are executed using the following parameter specifications (Parameter values were specified from preliminary experiments):

Population size: 200 strings,
Crossover probability: 0.8 (uniform crossover),
Biased mutation probabilities:
$p_{\mathrm{m}}(0 \to 1) = 1/300M$ and $p_{\mathrm{m}}(1 \to 0) = 0.1$,
Stopping condition: 5000 generations,
α and β in Fig. 3 and Fig. 4: $\alpha = 10$ and $\beta = 10$.

Each variant is applied to each data set 10 times. Table 2 shows the average number of obtained non-dominated rule sets over 10 runs. The largest value for each data set is highlighted by boldface in Table 2. The last row shows the average value over the six data sets for each variant. From Table 2, we can see that the removal of overlapping solutions by the first method increases the number of obtained non-dominated rule sets for many data sets. We also record the best error rate on training patterns among rule sets in the final population obtained by each variant for each data set. The average best error rate is calculated over 10 runs in Table 3. On the other hand, Table 4 shows the average error rate on training patterns of all the obtained non-dominated rule sets over 10 runs. We can see from Table 3 and Table 4 that the third method drove the population toward rule sets with high classification accuracy on training patterns.

TABLE 2. THE AVERAGE NUMBER OF OBTAINED RULE SETS BY EACH VARIANT OF THE NSGA-II ALGORITHM.

Data set	Original NSGA-II	Removal of overlap	Similar	Extreme & similar
Breast W	11.2	**14.2**	12.3	11.6
Diabetes	19.5	**23.3**	20.3	15.9
Glass	28.3	31.0	**32.4**	16.7
Heart C	45.5	**46.3**	46.0	15.0
Sonar	12.7	**15.8**	12.8	10.1
Wine	**14.2**	13.9	12.9	13.2
Average	21.9	**24.1**	22.8	13.8

TABLE 3. THE AVERAGE VALUE OF THE BEST ERROR RATES OBTAINED BY EACH VARIANT OVER 10 RUNS.

Data set	Original NSGA-II	Removal of overlap	Similar	Extreme & similar
Breast W	1.9	**1.7**	1.9	1.8
Diabetes	21.9	**21.8**	21.9	21.9
Glass	21.9	21.5	21.0	**20.6**
Heart C	29.1	28.9	29.0	**28.6**
Sonar	11.1	**10.0**	11.0	10.3
Wine	**0.0**	**0.0**	**0.0**	**0.0**
Average	14.3	14.0	14.1	**13.9**

TABLE 4. THE AVERAGE VALUE OF THE AVERAGE ERROR RATES OF OBTAINED NON-DOMINATED RULE SETS OVER 10 RUNS.

Data set	Original NSGA-II	Removal of overlap	Similar	Extreme & similar
Breast W	14.8	12.1	13.5	**10.3**
Diabetes	28.2	**27.3**	28.0	27.5
Glass	36.7	35.5	35.7	**25.8**
Heart C	38.3	38.2	38.2	**31.5**
Sonar	24.9	22.3	24.7	**15.8**
Wine	20.0	19.8	21.4	**18.4**
Average	27.1	25.9	26.9	**21.5**

Examples of non-dominated rule sets obtained by each variant of the NSGA-II algorithm are shown in Figs. 5-7 for the Cleveland heart disease data set (i.e., Heart C). It should be noted that non-dominated rule sets in each figure were obtained by a single run of each variant. We can observe in these figures the existence of the clear tradeoff between the accuracy on training patterns and the complexity of fuzzy rule-based classification systems. We can also see from these figures (and Table 3 and Table 4) that each method increases the search ability of the NSGA-II algorithm to find fuzzy rule-based systems with high accuracy on training patterns.

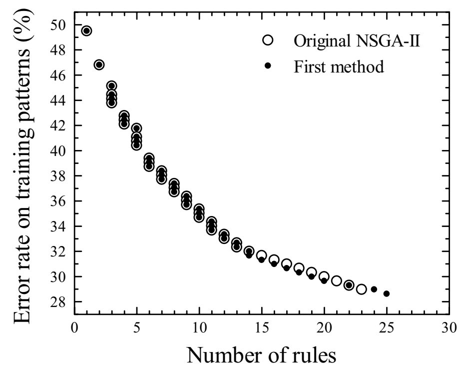

Fig. 5. Original NSGA-II and the first method.

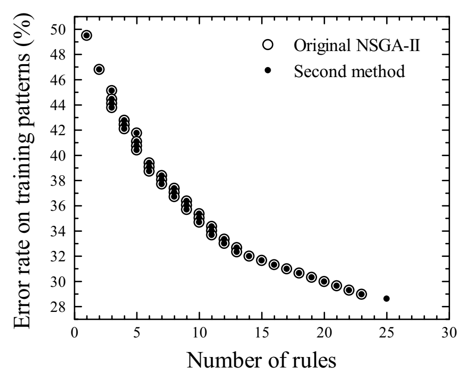

Fig. 6. Original NSGA-II and the second method.

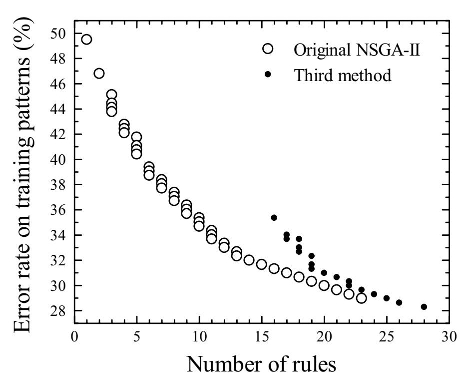

Fig. 7. Original NSGA-II and the third method.

V. CONCLUSIONS

In this paper, we examined three methods for improving the search ability of the NSGA-II algorithm to find a variety of non-dominated rule sets of a three-objective fuzzy rule selection problem. Experimental results showed that the performance of the NSGA-II algorithm was improved by removing overlapping solutions in terms of the variety of obtained non-dominated rule sets. It was also shown that the choice of extreme and similar parents drove the population toward rule sets with high classification accuracy.

ACKNOWLEDGMENT

The authors thank the financial support from the Okawa Foundation for Information and Telecommunications.

REFERENCES

[1] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, Chichester, 2001.

[2] C. A. Coello Coello, D. A. van Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer Academic Publishers, Boston, 2002.

[3] H. Ishibuchi, T. Murata, and I. B. Turksen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems," *Fuzzy Sets and Systems*, vol. 89, no. 2, pp. 135-150, 1997.

[4] H. Ishibuchi, T. Nakashima, and T. Murata, "Three-objective genetics-based machine learning for linguistic rule extraction," *Information Sciences*, vol. 136, no. 1-4, pp. 109-133, 2001.

[5] H. Ishibuchi and T. Yamamoto, "Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining," *Fuzzy Sets and Systems*, vol. 141, no. 1, pp. 59-88, 2004.

[6] H. Ishibuchi and T. Yamamoto, "Effects of three-objective genetic rule selection on the generalization ability of fuzzy rule-based systems," *Lecture Notes in Computer Science 2632: Evolutionary Multi-Criterion Optimization*, pp. 608-622, Springer, Berlin, 2003.

[7] H. Ishibuchi and T. Yamamoto, "Evolutionary multiobjective optimization for generating an ensemble of fuzzy rule-based classifiers," *Lecture Notes in Computer Science 2723: Genetic and Evolutionary Computation - GECCO 2003*, pp. 1077-1088, Springer, Berlin, July 2003.

[8] H. Ishibuchi and S. Namba, "Evolutionary multiobjective knowledge extraction for high-dimensional pattern classification problems," *Lecture Notes in Computer Science 3242: Parallel Problem Solving from Nature - PPSN VIII*, pp. 1123-1132, Springer, Berlin, 2004.

[9] H. A. Abbass, "Pareto neuro-evolution: Constructing ensemble of neural networks using multi-objective optimization," *Proc. of 2003 Congress on Evolutionary Computation*, pp. 2074-2080, Canberra, Australia, December 8-12, 2003.

[10] Y. Jin, T. Okabe, and B. Sendhoff, "Neural network regularization and ensembling using multi-objective evolutionary algorithms," *Proc. of 2004 Congress on Evolutionary Computation*, pp. 1-8, Portland, USA, June 19-23, 2004.

[11] C. Emmanouilidis, A. Hunter, and J. MacIntyre, "A multiobjective evolutionary setting for feature selection and a commonality-based crossover operator," *Proc. of 2000 Congress on Evolutionary Computation*, vol. 1, pp. 309-316, San Diego, USA, July 16-19, 2000.

[12] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "A methodology for feature selection using multiobjective genetic algorithms for handwritten digit string recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 17, no. 6, pp. 903-929, 2003.

[13] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, 1999.

[14] 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.

[15] H. Ishibuchi and Y. Shibata, "An empirical study on the effect of mating restriction on the search ability of EMO algorithms," *Lecture Notes in Computer Science 2632: Evolutionary Multi-Criterion Optimization*, pp. 433-447, Springer, Berlin, 2003.

[16] H. Ishibuchi and Y. Shibata, "A similarity-based mating scheme for evolutionary multiobjective optimization," *Lecture Notes in Computer Science 2723: Genetic and Evolutionary Computation - GECCO 2003*, pp. 1065-1076, Springer, Berlin, 2003.

[17] H. Ishibuchi and Y. Shibata, "Mating scheme for controlling the diversity-convergence balance for multiobjective optimization," *Lecture Notes in Computer Science 3102: Genetic and Evolutionary Computation - GECCO 2004*, pp. 1259-1271, Springer, Berlin, 2004.

[18] H. Ishibuchi and T. Nakashima, "Effect of rule weights in fuzzy rule-based classification systems," *IEEE Trans. on Fuzzy Systems*, vol. 9, no. 4, pp. 506-515, 2001.

[19] H. Ishibuchi, T. Nakashima, and M. Nii, *Classification and Modeling with Linguistic Information Granules*, Springer-Verlag, 2004.

[20] H. Ishibuchi and T. Yamamoto, "Comparison of heuristic criteria for fuzzy rule selection in classification problems," *Fuzzy Optimization and Decision Making*, vol. 3, no. 2, pp. 119-139, 2004.

[21] A. Gonzalez and R. Perez, "SLAVE: A genetic learning system based on an iterative approach," *IEEE Trans. on Fuzzy Systems*, vol. 7, no. 2, pp. 176-191, 1999.