Cellular Genetic Algorithm for Multi-Objective Optimization

Tadahiko Murata

Ashikaga Institute of Technology
murata@ashitech.ac.jp

Mitsuo Gen

Ashikaga Institute of Technology
gen@ashitech.ac.jp

Abstract

In this paper, we show how cellular structures can be combined with a multi-objective genetic algorithm (MOGA) for improving its search ability to find Pareto-optimal solutions of multi-objective optimization problems. We propose an assignment method of a different search direction to each cell for implementing a cellular MOGA. In our cellular MOGA, every individual in each population exists in a cell of a spatially structured space (e.g., two-dimensional grid-world) where each cell has a different search direction. Such a search direction corresponds to weights in a scalar fitness function defined by the weighted sum of multiple objectives. The selection of parents for generating a new individual in a cell is performed within the neighborhood of that cell based on its search direction. The effectiveness of the proposed cellular MOGA is shown by computer simulations on two-objective flowshop scheduling problems.

1 INTRODUCTION

Genetic algorithms have been successfully applied to various optimization problems [1]. The extension of GAs to multi-objective optimization was proposed in several manners (for example, Schaffer [2], Kursawe [3], Horn *et al.*[4], Fonseca & Fleming [5], Murata & Ishibuchi [6], and Zitzler & Thiele [7]). The aim of these algorithms is to find a set of Pareto-optimal solutions of a multi-objective optimization problem. Another issue in multi-objective optimization is to select a single final solution from Pareto-optimal solutions. Many studies on multi-objective GAs did not address this issue because the selection totally depends on the decision maker's preference. In this paper, we also concentrate our attention on the search for finding a set of Pareto-optimal solutions.

The concept of cellular genetic algorithms was proposed by Whitley [8]. In cellular genetic algorithms, each individual (i.e. a chromosome) resides in a cell of a spatially structured space. Genetic operations for generating new individuals are locally performed in the neighborhood of each cell. While the term "cellular genetic algorithm" was proposed by Whitley, such algorithms had already been proposed by Manderik and Spiessens [9]. A similar concept was also studied in evolutionary ecology in the framework of "structured demes" (Wilson [10], and Dugatkin & M. Mesterton-Gibbons [11]). The effect of spatial structures on the evolution of cooperative behavior has also been examined in many studies (e.g., Nowak & May [12], Wilson *et al.* [13], Oliphant [14], Grim [15], and Ishibuchi *et al.* [16]) where each individual was located in a cell of single-dimensional or two-dimensional grid-worlds.

In this paper, we extend the MOGA (Murata & Ishibuchi [6]) by combining the algorithm with cellular structures. That is, we assign every individual in each population to a cell in a spatially structured space (e.g., two-dimensional grid-world). We add the abbreviation "C-" to show "Cellular-" for the extended algorithms.

2 MULTI-OBJECTIVE OPTIMIZATION

Let us consider the following multi-objective optimization problem with n objectives:

$$\text{Maximize} \quad f_1(\boldsymbol{x}),\ f_2(\boldsymbol{x}), ...,\ f_n(\boldsymbol{x}), \tag{1}$$

where $f_1(\cdot)$, $f_2(\cdot)$, ..., $f_n(\cdot)$ are n objectives. We can assume that all functions are to be maximized without loss of generality. When the following inequalities hold between two solutions $\boldsymbol{x}$ and $\boldsymbol{y}$, the solution $\boldsymbol{y}$ is said to dominate the solution $\boldsymbol{x}$:

$$\forall i : f_i(\boldsymbol{x}) \le f_i(\boldsymbol{y}) \ \text{ and } \ \exists j : f_j(\boldsymbol{x}) < f_j(\boldsymbol{y}). \tag{2}$$

If a solution is not dominated by any other solutions of the multi-objective optimization problem, that solution is said to be a Pareto-optimal solution. The task of multi-objective algorithms in this paper is not to select a single final solution but to find all Pareto-optimal solutions of the multi-objective optimization problem in (1). When we use heuristic search algorithms such as taboo search, simulated annealing, and genetic algorithms for finding Pareto-optimal solutions, we usually can not confirm the optimality of obtained solutions. We only know that each of the obtained solutions is not dominated by any other solutions examined during the execution of those algorithms. Therefore obtained solutions by heuristic algorithms are referred to as "nondominated" solutions. For a large-scale multi-objective optimization problem, it is impossible to find all Pareto-optimal solutions. Thus our task is to find many near-optimal nondominated solutions in a practically acceptable computational time. The performance of different multi-objective algorithms is compared based on several quality measures of obtained nondominated solutions.

3 MULTI-OBJECTIVE GA

In this section, we explain the MOGA of Murata & Ishibuchi [6] that is extended in later sections of this paper. In the MOGA , the weighted sum of the n objectives is used as a fitness function:

$$f(\boldsymbol{x}) = w_1 f_1(\boldsymbol{x}) + w_2 f_2(\boldsymbol{x}) + \ldots + w_n f_n(\boldsymbol{x}), \qquad (3)$$

where $w_1, \ldots, w_n$ are nonnegative weights for the n objectives, which satisfy the following relations:

$$w_i \geq 0 \quad \text{for} \quad i = 1, 2, \ldots, n, \qquad (4)$$

$$w_1 + w_2 + \cdots + w_n = 1. \qquad (5)$$

This fitness function is utilized when a pair of parent solutions are selected for generating a new solution by crossover and mutation. One characteristic feature of the MOGA is to randomly specify weight values whenever a pair of parent solutions are selected. That is, each selection (i.e., the selection of two parents) is performed based on a different weight vector. This means that each of newly generated solutions by the genetic operations has its own weight vector. Another characteristic feature is to store two different sets of solutions: a current population and a tentative set of nondominated solutions. Because of this storing, any nondominated solution found during the execution of the algorithm is kept in the tentative set of nondominated solutions.

3.1 SELECTION OPERATION

When a pair of parent solutions are to be selected from a current population for generating an offspring by genetic operations, first the n weight values ($w_1, w_2, \ldots, w_n$) are randomly specified as follows:

$$w_i = random_i / (random_1 + \cdots + random_n), \quad i = 1, 2, \ldots, n, \qquad (6)$$

where $random_i$ are nonnegative random real numbers. For example, when N pairs of parent solutions are selected for generating a new population, N different weight vectors are specified by (6). This means that N search directions are utilized in a single generation. In other words, each selection (i.e., the selection of two parents) is governed by a different fitness function.

3.2 ELITIST STRATEGY

The MOGA separately stores two different sets of solutions: a current population and a tentative set of nondominated solutions. After the genetic operations are applied to each solution, the current population is replaced with the newly generated solutions. The tentative set of nondominated solutions is also updated by the generated solutions. That is, if a solution obtained by the genetic operations is not dominated by any other solutions in the newly generated current population and the tentative set of nondominated solutions, this solution is added to the tentative set. Then all solutions dominated by the added one are removed from the tentative set. In this manner, the tentative set of nondominated solutions is updated at every generation in the MOGA.

From the tentative set of nondominated solutions, a few solutions are randomly selected and added to the current population (see Figure 1). The randomly selected nondominated solutions may be viewed as elite solutions because they are added to the current population with no genetic operations.

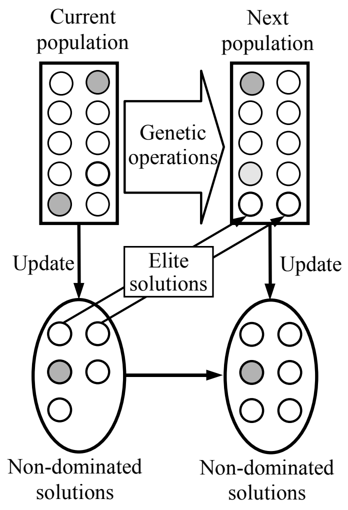

Figure 1: Illustration of the MOGA.

3.3 ALGORITHM

Let us denote the population size by N_{pop}. We also denote the number of nondominated solutions added to the current population by N_{elite} (i.e., N_{elite} is the number of elite solutions, see Figure 1). Using these notations, the procedures of the MOGA can be written as follows.

Step 0) Initialization: Randomly generate an initial population of N_{pop} solutions.

Step 1) Evaluation: Calculate the values of the n objectives for each solution in the current population. Then update the tentative set of nondominated solutions.

Step 2) Selection: Repeat the following procedures to select ($N_{\text{pop}} - N_{\text{elite}}$) pairs of parent solutions.

a) Randomly specify the weight values w_1, $w_2, \ldots, w_n$ in the fitness function (3) by (6).

b) According to the following selection probability $P(\boldsymbol{x})$, select a pair of parent solutions from the current population Ψ.

$$P(\boldsymbol{x}) = \frac{f(\boldsymbol{x}) - f_{\min}(\Psi)}{\sum_{\boldsymbol{x}\in\Psi}\{f(\boldsymbol{x}) - f_{\min}(\Psi)\}}, \qquad (7)$$

where $f_{\min}(\Psi)$ is the minimum fitness value in the current population Ψ.

Step 3) Crossover and Mutation: Apply a crossover operator to each of the selected ($N_{\text{pop}} - N_{\text{elite}}$) pairs of parent solutions. A new solution is generated from each pair of parent solutions. Then apply a mutation operator to the generated new solutions.

Step 4) Elitist Strategy: Randomly select N_{elite} solutions from the tentative set of nondominated solutions, and add the selected N_{elite} solutions to the ($N_{\text{pop}} - N_{\text{elite}}$) solutions generated in Step 3 to construct a population of N_{pop} solutions.

Step 5) Termination Test: If a prespecified stopping condition is satisfied, end the algorithm. Otherwise, return to Step 1.

4 CELLULAR ALGORITHMS

4.1 CELLS WITH WEIGHT VECTORS

In cellular algorithms, each individual resides in a cell in a spatially structured space (e.g., two-dimensional grid-world). For utilizing a cellular structure in the MOGA, we assign a different weight vector to each cell. That is, each cell has its own weight vector. For our n-objective optimization problem, cells are structured in an n-dimensional weight space. Figure 2 shows an example of structured cells for a two-dimensional optimization problem where the fitness function $f(\boldsymbol{x})$ is defined by two weights w_1 and w_2 as $f(\boldsymbol{x}) = w_1 f_1(\boldsymbol{x}) + w_2 f_2(\boldsymbol{x})$. In this figure, the population size is ten because an individual exists in each cell. As shown in Figure 2, the location of each cell corresponds to its weight vector. Weight vectors are uniformly specified according to the population size. For example, weight vectors for 11 individuals (i.e., for 11 cells) are (1.0, 0.0), (0.9, 0.1), ..., (0.0, 1.0).

4.2 DEFINITION OF NEIGHBORHOOD

We can intuitively define the neighborhood structure among cells. That is, we can utilize any distance between cells in the n-dimensional space in which cells are structured. For example, the Euclid distance can be used for measuring the distance between cells. In this paper, the nearest k_{neighbor} cells (including that cell itself with zero distance) from each cell are considered as its neighbors.

4.3 SELECTION

In the MOGA described in Section 3, each solution has its own weight vector, which was used for selecting its parent solutions. In our cellular multi-objective genetic algorithm (C-MOGA) proposed in this paper, however, each cell has its own weight vector. As shown in Figure 2, the weight vector assigned to each cell corresponds to its location. For generating a new individual for a cell by the genetic operations, we use its weight vector in the fitness function.

Two parents for generating a new individual in a cell are selected from its k_{neighbor} neighbors (including that cell itself). The fitness value of each neighbor is recalculated based on the weight vector assigned to the cell for which a new individual is generated. That is, each individual is differently evaluated by this recalculation procedure of the fitness function in the selection for each cell.

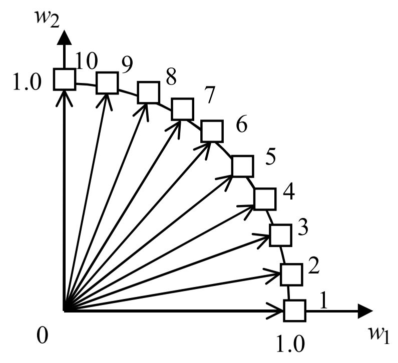

Figure 2: Location of each cell in the weight space.

5 COMPUTER SIMULATIONS

5.1 TEST PROBLEMS

We applied the proposed C-MOGA to flowshop scheduling problems. Flowshop scheduling is one of the most well-known scheduling problems. Since Johnson's work [17], various scheduling criteria have been considered. Among them are makespan, maximum tardiness, total tardiness, and total flowtime. Several researchers extended single-objective flowshop scheduling problems to multi-objective problems (see, for example, Daniels & Chambers [18]).

In this paper, we use the makespan and the total tardiness as two scheduling criteria in our flowshop scheduling problems. The makespan is the maximum completion time of all jobs to be processed. The total tardiness is the total overdue of all jobs. Let $g_1(\boldsymbol{x})$ and $g_2(\boldsymbol{x})$ be the makespan and the total tardiness. Since these scheduling criteria are to be minimized, we specify the two objectives $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ of our flowshop scheduling as $f_1(\boldsymbol{x}) = -g_1(\boldsymbol{x})$

and $f_2(\boldsymbol{x}) = -g_2(\boldsymbol{x})$.

As test problems, we generated ten 20-job and 10-machine flowshop scheduling problems. The processing time of each job on each machine was specified as a random integer in the interval [1, 99], and the duedate of each job was defined randomly. Our task is to find a set of Pareto-optimal solutions of each test problem. In our computer simulations, each solution $\boldsymbol{x}$ was represented by a permutation of 20 jobs.

5.2 QUALITY MEASURES OF SOLUTION SETS

Since multi-objective algorithms find a set of nondominated solutions with respect to multiple objectives (not a single final solution with respect to a single objective), the comparison between different multi-objective algorithms is not easy. For this purpose, we use the following measures for evaluating the quality of a solution set obtained by each algorithm.

1) **The number of obtained nondominated solutions**
The number of nondominated solutions obtained by each algorithm is a measure to evaluate the variety of the solution set.

2) **The number of solutions that are not dominated by other solution sets**
For comparing different solution sets with one another, we examine whether each solution is dominated by any other solutions in other sets. If a solution is dominated by another solution, we remove that solution. In this manner, we remove solutions dominated by other solution sets. The number of remaining solutions in each solution set is a measure for evaluating its relative quality with respect to the other solution sets.

3) **Set quality measure proposed by Esbensen [19]**
Esbensen [19] proposed an evaluation method of the quality of a solution set. Let us denote a solution set by Ω. The best solution $\boldsymbol{x}^*$ for a given weight vector $\boldsymbol{w} = (w_1, w_2)$ can be chosen from Ω for the two-objective optimization problem as follows:

$$\begin{aligned} f(\boldsymbol{x}^*) &= w_1 f_1(\boldsymbol{x}^*) + w_2 f_2(\boldsymbol{x}^*) \\ &= \max\{w_1 f_1(\boldsymbol{x}) + w_2 f_2(\boldsymbol{x}) \mid \boldsymbol{x} \in \Omega\}. \end{aligned} \tag{8}$$

Esbensen proposed an idea of measuring the quality of the solution set Ω by calculating the expected value of $f(\boldsymbol{x}^*)$ over possible weight vectors. In this paper, we calculate the expected value of $f(\boldsymbol{x}^*)$ by randomly generating 10,000 weight vectors by (6). That is, the quality of the solution set Ω is calculated as follows:

$$q(\Omega) = \frac{1}{10000} \sum_{i=1}^{10000} \max\{w_1^i f_1(\boldsymbol{x}) + w_2^i f_2(\boldsymbol{x}) \mid \boldsymbol{x} \in \Omega\}, \tag{9}$$

where $q(\Omega)$ is the quality of the solution set Ω and $\boldsymbol{w}^i = (w_1^i, w_2^i)$, $i = 1, 2, \ldots, 10000$ are randomly specified weight vectors.

4) **Maximum distance between two solutions**
The maximum distance between two solutions in the solution set shows its variety (i.e., the spread of solutions in the multi-dimensional objective space). This is defined for a solution set Ω as

$$D(\Omega) = \max\left\{ \sqrt{\sum_{i=1}^{n} (f_i(\boldsymbol{x}) - f_i(\boldsymbol{y}))} \mid \boldsymbol{x}, \boldsymbol{y} \in \Omega \right\}. \tag{10}$$

5.3 SIMULATION RESULTS

In order to show the effectiveness of the proposed cellular MOGA, we also examined the performance of the MOGA. In our computer simulations, we employed the following parameter specifications in each algorithm:

Population size: $N_{\text{pop}} = 100$ (i.e., 100 cells in C-MOGA),
Crossover: Two-point order crossover
(crossover rate: 0.8),
Mutation: Shift mutation (mutation rate: 0.3),
Number of elite solutions: $N_{\text{elite}} = 3$,
Neighborhood structure for the local search: Shift,
The number of neighboring cells in C-MOGA:
$k_{\text{neigh}} = 6, 10, 14, 20, 40$
Stopping condition: Examination of 50,000 solutions.

We used the above stopping condition in order to compare different algorithms under the same computation load. In a single trial of each algorithm, 50,000 solutions were examined. Since the population size was 100, we used 100 cells in the C-MOGA. The weight vectors of these cells were specified as $\boldsymbol{w} = (w_1, w_2) = (1.00, 0.00), (0.99, 0.01), \ldots, (0.00, 1.00)$.

We examined the effect of the introduction of the cellular structure (i.e., the locally restricted genetic operations). We compared the obtained set of nondominated solutions by the MOGA with that by the C-MOGA with $k_{\text{neighbor}} = 10$.

In Table 1, we summarize the average results over 100 trials for each algorithm (i.e. 10 trials for each of 10 test problems). In this table, “A” is the number of nondominated solutions obtained by each algorithm, and “B” is the number of solutions that are not dominated by other solutions obtained by the other algorithm. The ratio of these two numbers is shown in the column of B/A. “Quality” is the set quality measure of Esbensen, and “D” is the maximum Euclid distance between two solutions in the obtained solution set by each algorithm.

From Table 1, we can see that most solutions obtained by the MOGA are dominated by solutions obtained by the C-MOGA since the values of “B/A” and “Quality” obtained by the C-MOGA are better than those obtained by the MOGA. On the other hand, the maximum distance of the MOGA is better than that of the C-MOGA. The spread of the obtained solution set of the C-MOGA can be improved, but it should be noted that the maximum distance between two solutions in the obtained solution set can be large in a worse set of nondominated solutions. Thus we can conclude that the C-MOGA outperformed the MOGA.

Table 2 shows the average results over 100 trials for each specification of k_{neighbor} in the C-MOGA. From this table, we can see that the performance of the C-MOGA is not sensitive to the specification of k_{neighbor}. We specified k_{neighbor} as $k_{\text{neighbor}} = 10$ in the experiments in Table 1.

Table 1: Comparison of MOGA with C-MOGA.

	A	B	B/A	Quality	D
MOGA	14.6	4.2	0.296	-1065.2	1512.4
C-MOGA	15.9	13.6	0.857	-990.0	1342.2

Table 2: Effect of the choice of k_{neighbor} in C-MOGA.

	6	10	14	20	40
Quality	-1004.4	-990.0	-986.8	-985.8	-986.9

6 CONCLUSION

In this paper, we proposed a cellular multi-objective genetic algorithm (C-MOGA), which was an extension of a multi-objective genetic algorithm (MOGA) in Murata & Ishibuchi [6]. In the proposed C-MOGA, each individual was located in a cell with a different weight vector. This weight vector governed the selection operation at that cell. The selection was performed in the neighborhood of each cell. The effectiveness of the proposed algorithm was demonstrated by computer simulations on two-objective flowshop scheduling problems. The implementation of the proposed algorithms for optimization problems with three or more objectives is left for future work.

REFERENCES

[1] D.E.Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[2] J.D.Schaffer. Multi-objective optimization with vector evaluated genetic algorithms. *Proc. of 1st International Conference on Genetic Algorithms*: pp. 93-100, 1985.

[3] F.Kursawe. A variant of evolution strategies for vector optimization. In H.-P.Schwefel and R.Männer (Eds.), *Parallel Problem Solving from Nature*. 193-197. Berlin: Springer-Verlag, 1991.

[4] J.Horn, N.Nafpliotis and D.E.Goldberg. A niched Pareto genetic algorithm for multi-objective optimization. *Proc. of 1st IEEE International Conference on Evolutionary Computation*: pp. 82-87, 1994.

[5] C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization, *Evolutionary Computation* 3: pp. 1-16, 1995.

[6] T.Murata and H.Ishibuchi. MOGA: Multi-objective genetic algorithms. *Proc. of 2nd IEEE International Conference on Evolutionary Computing*: pp. 289-294, 1995.

[7] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto Approach. *IEEE Trans. on Evolutionary Computation* 3: pp. 257-271, 1999.

[8] D. Whitley. Cellular Genetic Algorithms. *Proc. of 5th International Conference on Genetic Algorithms*: page 658, 1993.

[9] B.Manderick and P.Spiessens. Fine-grained parallel genetic algorithms. *Proc. of 3rd International Conference on Genetic Algorithms*, 1989.

[10] D. S. Wilson (1977). Structured demes and the evolution of group-advantageous traits. *The American Naturalist*, **111** (977): 157-185.

[11] L. A. Dugatkin and M. Mesterton-Gibbons. Cooperation among unrelated individuals: Reciprocal altruism, by-product mutualism and group selection in fishes. *BioSystems*, **37**: pp. 19-30, 1996.

[12] M. A. Nowak and M. May. Evolutionary games and spatial chaos. *Nature*, **359**: pp. 826-859, 1992.

[13] D. S. Wilson, G. B. Pollock, and L. A. Dugatkin. Can altruism evolve in purely viscous populations? *Evolutionary Ecology*, **6**: pp. 331-341, 1992.

[14] M. Oliphant. Evolving cooperation in the non-iterated Prisoner's Dilemma: The importance of spatial organization. in R. A. Brooks and P. Maes (eds.), *Artificial Life IV* (MIT Press, Cambridge): pp. 349-352, 1994.

[15] P. Grim. Spatialization and greater generosity in the stochastic Prisoner's Dilemma. *BioSystems*, **37**: pp. 3-17, 1996.

[16] H. Ishibuchi, T. Nakari, and T. Nakashima. Evolution of neighborly relations in a spatial IPD game with cooperative players and hostile players. *Proc. of Congress on Evolutionary Computation*: pp. 929-936, 1999.

[17] S.M.Johnson. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly* **1** (1): pp. 61-68, 1954.

[18] R.L.Daniels and R.J.Chambers. Multiobjective flow-shop scheduling. *Naval Research Logistics* **37**: pp. 981-995, 1990.

[19] H.Esbensen. Defining solution set quality. *Memorandum* (No.UCB/ERL M96/1, Electric Research Laboratory, College of Engineering, University of

California, Berkeley, USA, January, 1996).