Genetic Algorithm with a Pareto Partitioning Method for Multiobjective Flowshop Scheduling

Takanori Tagami† and Tohru Kawabe‡

†*Dept. of Management and Information Processing, Kagawa Junior College*
Kagawa 769-0201 Japan, tagami@kagawa-jc.ac.jp
‡*Institute of Information Sciences and Electronics, University of Tsukuba*
Ibaraki 305-8573 Japan, kawabe@is.tsukuba.ac.jp

Abstract— This paper describes a genetic algorithm using a Pareto partitioning method for multiobjective flowshop scheduling. The purpose of the proposed method is to generate a set of non-dominated solutions that is properly distributed in the neighborhood of the trade-off surface. Simulation results show that the GA using the Pareto partitioning method has good performances better than the traditional GA approaches for several 2-objective flowshop scheduling problems.

I. INTRODUCTION

Many real-world problems require the simultaneous optimization of multiple, often competing, objectives, the solution to such problems is usually computed by combining them into a single objective to be optimized, according to some utility function. In many cases, the utility function is not well known prior to the optimization process. The whole problem should then be treated as a multiobjective problem with noncommensurable objectives. In this way, a number of solutions can be found which provide the decision maker (DM) with insight into the characteristics of the problem before a final solution is chosen.

Multiobjective optimization seeks to optimize the components of a vector-valued cost function. Unlike single objective optimization, the solution to this problem is not a single point, but a family of points known as the Pareto-optimal set, also called non-dominated set. Each point in the trade-off surface is optimal in the sense that no improvement can be achieved in one cost vector component that does not lead to degradation in at least one of the remaining components. Conventional optimization techniques, such as gradient-based method, simplex-based method and simulated annealing, are difficult to extend to the true multiobjective optimization case, because they were not designed with multiple solutions in mind.

Genetic algorithms (GAs)[1,2] have been recognized to be well suited to multiobjective optimization since early in their development. Multiple individuals can search for multiple solutions in parallel, eventually taking advantage of any similarities available in the family of possible solutions to the problem. Extensions of GAs to multiobjective optimization were proposed in several manners [3,4]. Schaffer [3] proposed an extension of the simple GA to accommodate vector-valued fitness measures, which he called the Vector Evaluated Genetic Algorithm (VEGA). In VEGA, appropriate fractions of the next generation, or sub-populations, were selected from the whole of the old generation according to each of the objective values, separately. However, in the case of a concave trade-off surface, the population will tend to split into different species, each of them particularly strong in one of the objectives. This phenomenon, known as speciation, is undesirable in that it is opposed to the aim of finding a compromise solution.

While a Pareto-based approach was first proposed by Goldberg [1], as a means of assigning equal probability of reproduction to all non-dominated individuals in the population. Fonseca and Fleming [4] proposed a multiobjective ranking based on the Pareto-based fitness assignment. In this method, an individual's rank corresponds to the number of individuals in the current population by which it is dominated. Multiobjective ranking correctly assigns all non-dominated individuals the same fitness, but that, on its own, does not guarantee that the Pareto-optimal set be uniformly sampled. When presented with multiple equivalent solutions, finite population member tend to converge to only one of them, due to stochastic errors in the selection process. This phenomenon, known as genetic drift, has been observed in natural as well as in artificial evolution, and can also occur in the Pareto-based evolutionary optimization.

In this paper, a genetic algorithm using a Pareto partitioning method to multiobjective optimization problems is proposed. The purpose of the proposed

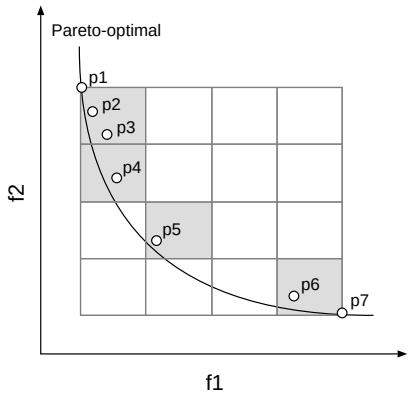

Figure 1: Pareto partitioning method in two objective minimizing problem.

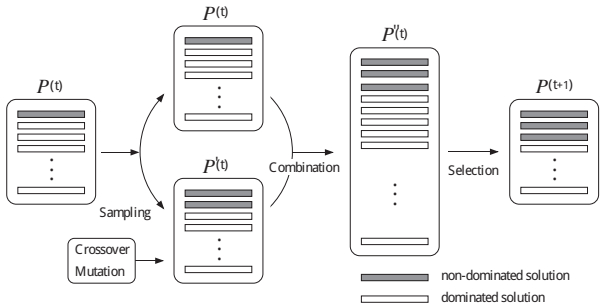

Figure 2: Generation model with a Pareto-optimal preservation strategy.

method is to generate a Pareto-optimal set that is properly distributed in the neighborhood of the trade-off surface. The genetic search using the proposed method uniformly control the convergence of solutions.

Experimental results for several flowshop scheduling problems indicate that the standard GA with the multiobjective ranking leads to the partial convergence of the solutions. On the other hand, the GA with the proposed methods are effective in directing the genetic search in the objective space.

II. PROPOSED ALGORITHM

A. Pareto partitioning method

To prevent a partial convergence of non-dominated solutions in the trade-off surface, we propose the Pareto partitioning method which uniformly controls a distribution of non-dominated solutions. The proposed method assigns all non-dominated individuals the differ fitness on the basis of the distribution in pre-specified regions. An example of the Pareto partitioning method in two objective minimizing problem is shown in Figure 1.

The proposed method consists of following procedure. First, the objective space is divided pre-specified regions. The edge points of the whole region correspond the best solutions for each objective function. In the Figure 1, the individuals p_1 and p_7 match them. Then, the fitness f_i of the individual p_i is defined as $f_i = 1/n_i$. The value of n_i denotes the number of non-dominated solutions in the identical region with the individual p_i. In the example, the fitness of the individuals illustrated in the figure correspond to the following values $(f_1, f_2, f_3, f_4, f_5, f_6, f_7) = (1/3, 1/3, 1/3, 1, 1, 1/2, 1/2)$.

B. Proposed procedure

The continuous generation model with a Pareto-optimal preservation strategy is employed, and is illustrated in Figure 2. In this model, a Pareto-optimal set is always forced to appear in the following generation.

The proposed procedure consists of the following steps:

Step 1 Set a generation number $t = 0$. Randomly generate an initial population $P(t)$ of M individuals.

Step 2 Calculate the fitness of each solution in the current population according to the multiobjective ranking. If the non-dominated solutions occupy the current population, apply the Pareto partitioning method for the fitness assignment of all non-dominated individuals.

Step 3 Generate a new population $P'(t)$ from $P(t)$ by using a crossover operator.

Step 4 Apply a mutation operator to the newly generated population $P'(t)$ according to mutation rate.

Step 5 Calculate the fitness both of $P(t)$ and $P'(t)$.

Step 6 Select M individuals from all population member on the basis of the fitness. If the non-dominated solutions is over M, select M individuals from whole population according to the proposed fitness assignment.

Step 7 If a terminal condition is satisfied, stop and return the best individuals. Otherwise set $t = t + 1$ and go to **Step 2**.

In this procedure, update of the current population size is always constant M. There is no question that selection of individuals for the next generation should be proportional to their fitness values. However, when a pure proportional approach is adopted, in practice the population is rapidly dominated by a few super-individuals, resulting in a dramatic decrease in genetic diversity. Here, to avoid the rapid loss of genetic diversity, multiple equivalent individuals are eliminated from the current population.

III. COMPUTER SIMULATIONS

A. Problem statement

Flowshop scheduling is one of the most well-known scheduling problems. Several scheduling criteria have been examined in the literature. In this paper, we use two criteria (makespan and maximum lateness) in order to allow visualization. We specify two objective minimizing problem as follows:

$$\text{minimize} \quad F(\boldsymbol{x}) = \big(F_1(\boldsymbol{x}), F_2(\boldsymbol{x})\big) \tag{1}$$

$$F_1(\boldsymbol{x}) = \max_j C_j \tag{2}$$

$$F_2(\boldsymbol{x}) = \max_j L_j \tag{3}$$

where $\boldsymbol{x}$ is a feasible solution of the flowshop scheduling problem. C_j denotes the completion time of job j and L_j denotes the lateness of job j.

In computer simulations, we generate 20-job, 40-job and 50-job scheduling problems with 10-machine. Thus, our scheduling problem is to determine the sequence of given jobs that are processed on 10 machines. The processing time of each job on each machine is specified as a random integer in the interval $[1, 100]$.

The duedate of each job are specified through three steps. First, randomly generate a sequence of the given jobs. Secondly, calculate the completion time of each job when the given jobs are processed in the job sequence above. Finally, define the duedate of each job by

$$d_j = C_j + rand[-200, 200], \tag{4}$$

where d_j is the duedate of job j, and $rand[-200, 200]$ is a random integer in the interval $[-200, 200]$.

B. GA formulation

To design a GA for multiobjective optimization problem, certain problem-dependent algorithm elements need to be defined. They influence both the efficiency of the algorithm and the quality of its results.

B.1. Representation of individuals

Consider a k-job scheduling problem. Each individual p_i is defined as an integer sequence whose elements are in the range $[1, \ldots, k]$, in such a way that the individual denotes the permutation of job number. In this way, each individual represents a possible scheduling configuration (a candidate solution).

B.2. Fitness function

The linear scaling method based on the multiobjective ranking is employed as a fitness function. The fitness f_i of the individual p_i is computed through two steps. First, the rank R_i $(i = 1, \ldots, M)$ of each individual in the current population $P(t)$ is calculated by the multiobjective ranking. Then, the fitness f_i of the individual p_i is defined as follows:

$$f_i = (f_{\max} - f_s \times (R_i - 1)) \tag{5}$$

$$f_s = \frac{1}{M-1}(f_{\max} - f_{\min}) \tag{6}$$

where M denotes the population size, $f_{\max}$ and $f_{\min}$ denote the maximum and the minimum fitness values. In computer simulations, the values of ($f_{\max}$, $f_{\min}$) are held constants (10.0, 1.0), namely, the fitness of best individuals is 10.0, on the contrary, the fitness of worst one is 1.0.

B.3. Genetic operators

The genetic operators applied in the algorithm are partial match crossover, reverse mutation and roulette wheel selection. Partial match crossover generates two offspring by exchanging a predefined elements of alternate subsections between two parent strings. The recombination operator, mutation, is implemented as reversing sequence at randomly selected string elements.

B.4. Control parameters

During all experiments, the population size is set as 50, the mutation rate is set as 0.1. The stopping condition is set as 500 generations at 20-job problem, and 1000 generations at 40-job and 50-job problems respectively. In the Pareto partitioning method, the number of partitioning region is set as $50^2 = 2500$.

C. Simulation results

Three GA approaches are used of comparison. The GA with the Pareto partitioning method is called GA1. The standard GA with multiobjective ranking is called GA2. The standard GA with sharing method is called GA3. In GA2, the newly generated population member is randomly selected from current population when the number of non-dominated solutions is over M. The genetic operators of GA2 and GA3 are same as GA1. In the simulations, search performance is also compared with the Interative Improvement Method (IIM).

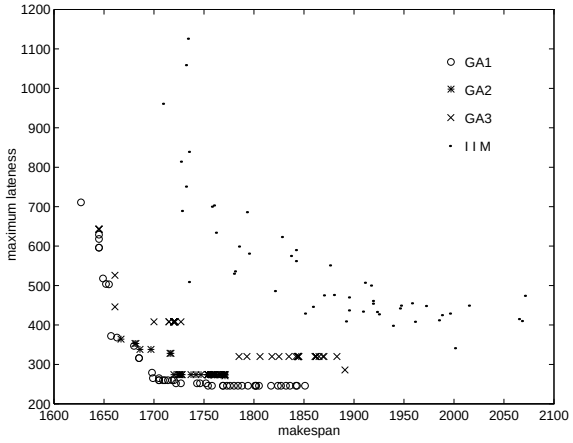

Figure 3: Distribution of solutions for the flowshop scheduling problem,(20 jobs)

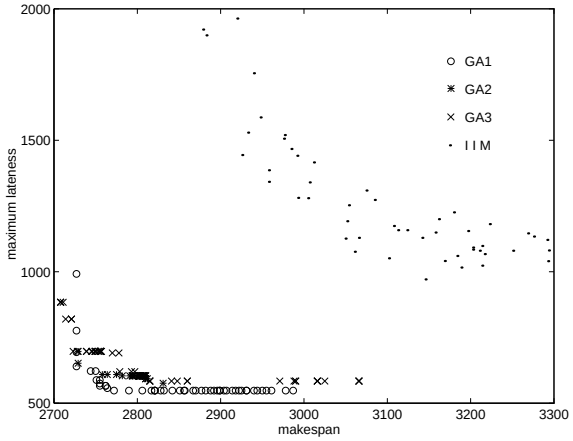

Figure 4: Distribution of solutions for the flowshop scheduling problem,(40 jobs)

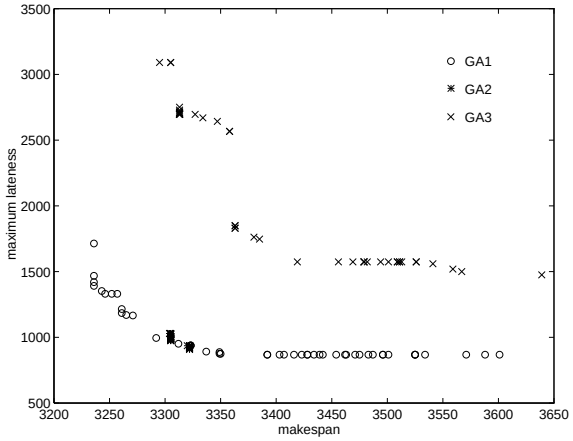

Figure 5: Distribution of solutions for the flowshop scheduling problem,(50 jobs)

The simulation results are shown in Figure 3, 4 and 5 for the cases when the number of jobs is 20, 40 and 50, respectively. In all figures, small marks indicate the solutions in the final population.

The following results are obtained:

- By using the proposed method, the solutions are widely distributed in the trade-off surface, and the search performance does not deteriorate significantly even in the all cases.
- From the results of GA2, the conventional Pareto-based approaches cause the partial convergence of the solutions because of stochastic errors in the iterative process.
- The performance of GA3 decays gradually as the number of job increases. The reason for this is a decrease of selective pressure in genetic search.
- It is clear that the proposed method can seek for the widely distributed solutions in comparison with the standard GA approaches and the interative improvement method.

IV. CONCLUSION

In this paper, we have proposed a genetic algorithm using a Pareto partitioning method in order to avoid the partial convergence of the solutions. The proposed Pareto partitioning method uniformly controls a distribution of non-dominated solutions in the objective space. We can confirm from several numerical simulations that the proposed method is effective for the generation of a Pareto-optimal set that is widely distributed in the trade-off surface.

Our next work will be the application of the present operation to various multiobjective optimization problems to investigate its general effectiveness.

References

[1] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine learning," *Addison Wesley*, 1989.

[2] Z. Michalewics and C. Janikow, "Genetic Algorithms + Data Structures = Evolution Programs," *Springer Verlag*, 1992.

[3] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," *Proc. of the 1st ICGA*, pp. 93–100, 1985.

[4] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: Formulation, Discussion and Generalization," *Proc. of the 5th ICGA*, pp. 416–423, 1993.

[5] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary Computation*, vol. 3, no. 1, pp. 1–16, 1995.