# Application of the Partial Enumeration Selection Method in Genetic Algorithms to Solving a Multi-Objective Flowshop Problem

Yong Zhao, Carlos A. Brizuela* and Nobuo Sannomiya
*Kyoto Institute of Technology, Matsugasaki, Sakyo-ku,*
*Kyoto 606–8585, Japan, {zhaoy, sanmiya}@si.dj.kit.ac.jp*
**Centro Nacional de Computación, Universidad Nacional de Asunción,*
*Campus Universitario, Paraguay, carlos.brizuela@sce.cnc.una.py*

## Abstract

In this paper, a partial enumeration selection method (PESM) is adopted in a genetic algorithm for solving a multi-objective flowshop problem. Based on the idea of adjusting selection pressure and reinforcing the Pareto front, the PESM-based genetic algorithm balances the exploitation and the exploration. Without the time-consuming computation of distance information among individuals and the hard-set parameters, this algorithm implicitly maintains diversity in the population. The PESM-based genetic algorithm is implemented and tested on a multi-objective flowshop scheduling problem. In order to compare the solution quality, an outperformance rate measure is proposed to work together with comparison of diversity. Simulation results show that the algorithm proposed improves specific results recently available in the literature and gets smooth non-dominated fronts.

## 1 Introduction

Most real-world optimization problems in manufacturing systems and economics are multi-objective. For these problems, where a set of Pareto solutions are more preferable than a single "best" solution, multi-objective genetic algorithms (MOGAs) are attracting researchers' attention because these algorithms maintain a large amount of solutions at a single run.

Keeping diversity is usually more important for MOGA than for a single objective genetic algorithm (SOGA) [1, 6, 7]. Various methods [1, 11] are proposed to promote uniform sampling and to control the distribution of individuals. Most of them define and calculate the genotype or phenotype distance among pairs of individuals. Parameters are also introduced. Without prior understanding of the problem landscape, these parameters are hard to be set [1].

In this paper, a partial enumeration selection method (PESM) is applied to maintain and control the diversity among the individuals in the genetic search. The PESM is originally proposed to solve single-objective jobshop scheduling problems [5] by adjusting the selection pressure. Without computation of distance information, diversity among individuals is maintained implicitly.

For each generation, the PESM samples a fixed number of individuals and enumerates some of their neighbors for selection of the individuals to replace the sampled ones. It gives us a chance to reinforce a Pareto front edge if a sampled individual lies in the current Pareto front. On the other side, if the PESM chooses not so good individuals, it may help to increase exploration. Furthermore, at each generation, the elitist individuals are preserved.

Lacking of a fair methodology for comparing results for a multi-objective problem restricts the development of MOGAs [8]. In this paper, a measure named outperformance rate is proposed to give us a clear numerical understanding. Together with comparison of diversity, we suggest a way to compare two MOGAs.

The genetic algorithm is implemented and tested on a multi-objective flowshop problem which first appeared in [6]. Simulation results show that the PESM improves the solution results easily and gets smooth Pareto fronts with high diversity.

This paper is organized as follows. After a statement of the multi-objective flowshop problem and a brief description of related works in sections 2 and 3, respectively, we describe the PESM in section 4. Finally, we present the definition of outperformance rate, report the simulation results, and give the conclusions in sections

0-7803-77-2/01/$10.00 

5, 6, and 7, respectively.

# 2 Multi-Objective Flowshop Problem

## 2.1 Problem statement

A permutation flowshop problem has $n$ jobs $(J_1, \ldots, J_n)$ that must be processed on a set of $m$ machines $(M_1, \ldots, M_m)$ with the same processing route. Each job $J_i$ consists of $m$ operations $O_{ij}(j = 1, ..., m)$ which must be processed on machine $M_j$ with processing time $p_{ij}$. A job can not be processed on different machines at the same time and no preemption is allowed. For each job $J_i$, a due date $d_i$ is assigned.

Assuming that all the job sequences on the machines are the same, which greatly decreases computational complexity without sacrificing solution quality, a feasible schedule $S$ is represented by a sequence of jobs:

$$\{j_1^S, \ldots, j_n^S\} \tag{1}$$

In order to define objectives, we denote the starting time and the finish time of job $J_i$ by $b_i$ and $c_i$, respectively. The objective functions concerned in [2, 6] and also in this paper, consisting of the makespan $(f_m)$, the mean tardiness $(f_t)$, and the mean flow time $(f_f)$, are given by the following equations:

$$\text{minimize}\,(f_m(S), f_t(S), f_f(S)) \tag{2}$$

where

$$f_m(S) = \max_i(c_i) \tag{3}$$

$$f_t(S) = (1/n)\sum_i(\max(c_i - d_i, 0)) \tag{4}$$

$$f_f(S) = (1/n)\sum_i(c_i - b_i) \tag{5}$$

The goal is to minimize these objectives as shown in Equation 2. This problem is very difficult since a flowshop scheduling problem solely with makespan as the objective to minimize has been proved to be NP-hard [3].

## 2.2 Pareto solutions

Because it is difficult to compare two solutions for a multi-objective problem, a set of solutions called non-dominated or Pareto solutions [7] are defined, where no objective can be improved without sacrificing any other. For the flowshop problem in this paper, a schedule $S_p$ is a Pareto solution if and only if there is not another feasible solution $S$ satisfying the following conditions:

$$f_m(S) \leq f_m(S_p), f_t(S) \leq f_t(S_p), f_f(S) \leq f_f(S_p), \tag{6}$$

$$f_m(S) + f_t(S) + f_f(S) < f_m(S_p) + f_t(S_p) + f_f(S_p) \tag{7}$$

# 3 Related work

## 3.1 Multi-objective genetic algorithm preliminaries

Since the first notable work of the Vector Evaluation Genetic Algorithm (VEGA), MOGAs have received considerable attention. The major difference between MOGA and SOGA lies on three points, the fitness assignment [7], the diversity control [1] and the effect of elitism [5].

First, in order to compare two solutions for a multi-objective problem, implicit Pareto ranking method is widely used in MOGAs. The ranking procedure is as follows: After Rank 0 is assigned to the non-dominated individuals they are removed from contention; then the next set of non-dominated individuals is assigned a Rank 1. The process is repeated until all individuals are ranked. Rank 0 is called the Pareto front.

Secondly, because of the phenomenon known as genetic drift, where individuals converge to only one solution, sharing techniques are proposed to promote uniform sampling. Goldberg and Richardson proposed and embedded niche-based fitness sharing in a non-dominated sorting genetic algorithm (NSGA). A crowd comparison criterion is proposed in NSGA II, an advanced version of NSGA, to eliminate the niching parameters. Furthermore, a coevolutionary niching method [4] is also proposed. For all these methods, they define distance and promote diversity based on distance information. Some methods, for example NSGA, introduce some parameters, which are difficult to be set without prior understanding of problem landscape.

Thirdly, many authors pointed out that saving the elitist individuals at each generation can greatly improve

the performance of MOGA. Therefore many elitist MOGAs, such as SPEA, PAES and NSGA-II are proposed [9].

### 3.2 Multi-objective flowshop problems

Comparing with many MOGAs for solving general multi-objective problems, research on multi-objective scheduling problems is rather scarce. This can be explained by the complex landscape and lacking of good methodology for the comparison of results [8].

In [6], the NSGA is presented and implemented to deal with flowshop and jobshop scheduling problems. By introducing elitism, Bagchi proposed the ENGA [6]. A detail of these algorithms and the tested problem data can be found in [6]. In [2], Isibuchi and Murata presented a dynamic-weighted local search without considering the non-dominated properties. Recently, Brizuela et al. [8] analyzed the genetic operators. With this analysis they proposed an extended NSGA to improve solution quality.

## 4 Partial Enumeration Selection Method

The partial enumeration selection method is originally proposed in solving single-objective jobshop scheduling problems [5]. The basic idea is simple in the sense that we do not need to compute the population diversity nor any other related measure. The mutation operator can be used to increase and decrease diversity. Adjusting the selection pressure may be another alternative. The PESM is focused on the latter issue. Sampling some individuals and enumerating some of their neighbors for selection of the individuals to replace the sampled ones is the way to control selection pressure and population diversity.

Assume that the offsprings directly replace their parents and selection of parents for genetic operations is performed at random, the selection pressure is null and we have a highly diverse population. Then the algorithm becomes a random search. This high diversity value will depend on the initial diversity of population and there will not be any possibility of changing it since the crossover and mutation operators will generate offsprings in a random way.

The PESM introduces order through the sorting of a sampled and enlarged set of individuals, and the selection of the best elements to take part in the genetic operations. Therefore, PESM increases the selection pressure and the search is no longer a random one. Figure 1 gives a flow chart of the PESM and algorithm 1 states it formally.

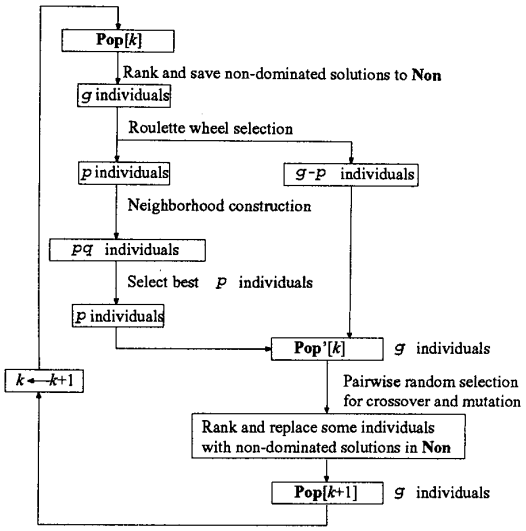



Figure 1: Partial Enumeration Selection Method.

**Algorithm 1. Partial Enumeration Selection Method (PESM)**

**Step 1:** Set $k = 0$. Generate an initial population **Pop**[$k$] of $g$ individuals.

**Step 2:** Ranking of individuals in **Pop**[$k$]. Generate the non-dominated individual pool **Non** consisting of all non-dominated individuals in **Pop**[$k$]. Assign fitness to individuals as the maximum rank number in the population minus the individual rank number.

**Step 3:** Set $i = 1$.

**Step 4:** Select one individual $I_i$ from **Pop**[$k$] by roulette wheel selection (RWS).

**Step 5:** Construct a neighborhood of $q$ elements around $I_i$ : $\{I_i, J1_i, \cdots, J(q-1)_i\}$.

**Step 6:** If $i \leq p$ then set $i = i + 1$ and go to Step 4; otherwise go to Step 7.

**Step 7:** From the new $pq$ individuals select the "best" $p$ elements according to the ranking criterion.

**Step 8:** Merge these $p$ elements with the $g-p$ individuals not selected at Steps 4 to 6 to get **Pop'**[$k$].

**Step 9:** Until $g$ individuals are chosen from **Pop**[$k$] DO: select randomly (with replacement) pairs of individuals and apply crossover and mutation with their respective probabilities.

**Step 10:** Rank all individuals in **Pop**[$k$] and the non-dominated individual pool **Non**. In **Non**, we have $nd$ non-dominated solutions.

**Step 11:** Select $nd$ individuals in **Pop'**[$k$] randomly and replace them with the $nd$ non-dominated solutions in **Non**; and obtain **Pop**[$k+1$]. Set $k = k+1$.

**Step 12:** If the stop criterion expires then stop, otherwise go to Step 2.

In the above algorithm, **Pop**[$k$] represents the set of $g$ individuals at generation $k$. Two positive numbers $p$ and $q$ stand for the number of individuals to sample from **Pop**[$k$] and the neighborhood size, respectively. At Step 5, for each sampled individual, we generate $q$ neighbors. Next at Step 7, the "best" $p$ individuals are selected from $pq$ members in the pool to replace the original $p$ individuals in **Pop**[$k$]. Noted that the "best" $p$ individuals are gotten by the following procedure: First these $pq$ individuals are ranked, then we choose the "best" individuals from Rank 0. If they are not enough (in number) to complete $p$ individuals, then we choose individuals from Rank 1. This procedure is repeated until we get $p$ individuals.

Notice that two issues in our proposed PESM extension are different from the PESM for SOGA [5]. First, all individuals in **Pop**[$k$] are ranked and sampled by Roulette Wheel Selection. The second issue is adoption of the elitism in PESM for multi-objective problems. This second issue reinforces the Pareto front. Since a large number of elitist individuals are inherited through generations, it is necessary to decrease their chance of being sampled. At Step 2, the fitness is assigned as the maximum rank number minus the individual ranking number. Therefore, the worse the individuals is, the higher the chance of being selected for partial enumeration of its neighbors. This process promotes exploitation.

The diversity can be controlled by the parameter $q$. Different values of $q$ will give different levels of selection pressure to the PESM. By increasing $q$, we increase the neighborhood size. Thus, we promote exploitation. In the extreme condition where $q = 1$, the PESM is a total random search algorithm except for the memory of the preserved elitist solutions. When we increase $q$, diversity begins to decrease and the PESM becomes more selective and better individuals are created.

# 5 A Measure for MOGAs

Unlike single-objective problems, comparing the solution quality of two MOGAs itself is a multi-objective problem. That is to say, we want a large amount of widely distributed Pareto solutions. In some cases, the Pareto fronts obtained by two MOGAs are plotted and we decide which one is better or worse by visual inspection. This can work effectively if the two MOGAs have steady performances. But, we usually want to have our decision based on many different runs of two algorithms. In this paper, a measure called outperformance rate is defined and it enables us to survey the results of multiple runs.

**Definition 1.** *Outperformance rate*: Let $F_1$ and $F_2$ be the Pareto fronts obtained by MOGA1 and MOGA2, repectively. Then we mix them and rank them. Next we count individuals in the new combined Pareto front $F_n$. Let $n_1$ be the number of individuals in the Pareto front $F_n$ which belong to $F_1$ and $n_2$ is the number belonging to $F_2$. Then $n_1/(n_1+n_2)$ is defined as the outperformance rate of MOGA1.

Another important issue is the diversity among individuals in the Pareto front. A Pareto front is bad if most of its individuals converge to few solutions. Therefore, we need to compare the diversity between populations. Diversity is defined as the average distance of all pairs of individuals. The distance measures the difference between two individuals. For the permutation flowshop problem, a phenotype distance is defined as the Euclidean distance of all objectives by:

$$dif(S_1,S_2) = ((f_m(S_1)-f_m(S_2))^2+(f_t(S_1)- f_t(S_2))^2+(f_f(S_1)-f_f(S_2))^2)^{1/2} \quad (8)$$

$$Diversity = 1/g(g-1)\sum_{ij,i\neq j} dif(S_i,S_j) \quad (9)$$

# 6 Experimental Setup and Results

A number of experiments are carried out in order to verify the performance of the PESM. A 49-jobs and 15 machines flowshop problem presented in [6] is tested in this paper.

First, we investigate the Pareto front and the outperformance rate by comparing PESM with a fine tuned elitist NSGA [8] and ENGA [6]. Secondly, we examine the variation of diversity among all individuals in the Pareto fronts.

The parameters of PESM and NSGA are set as follows: The population size $g$ is set to 100 and the algorithms stop after 2000 generations. Crossover rate $P_c$ is 1.0 and mutation rate $P_m$ is 0.05. Without special description, $p$ in PESM is set to 10 and $q$ is set to 10. Sharing parameter for NSGA is set to 10 which is considered as the best value for this problem by a large amount of simulations. The computation results are based on the average of 50 runs.

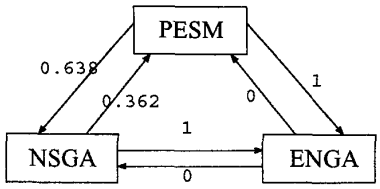
## 6.1 Comparison of Pareto fronts

Figure 2 shows outperformance rates among three different MOGAs. Clearly, NSGA and PESM outperform ENGA with 1.0 outperformance rate. Based on the 0.638/0.362 rate, PESM gets better result than NSGA.

Figure 3 plots a typical distribution of Pareto fronts of these MOGAs. Again, it is observed that PESM outperforms NSGA and NSGA outperforms ENGA. Figure 4 shows the dynamics of number of individuals in the Pareto fronts. NSGA has more non-dominated individuals than PESM. By analyzing Figure 4 together with the outperformance rate, it is observed that PESM has less but higher-quality individuals in the Pareto front.

## 6.2 Variation of diversity

As described in section 5, diversity is a very important issue for MOGAs. Figures 5 and 6 show the variation of diversity among all individuals and among individuals in the Pareto front, respectively. As expected, PESM has higher overall diversity. It is observed from Figure 6 that PESM maintains high diversity for all generations while NSGA gets equivalent high diversity from the 1500th generation.




$A \underset{b}{\overset{a}{\rightleftarrows}} B$ means that a and b are the outperformance rates of A and B, respectively

Figure 2: Outperformance rates among three MOGAs

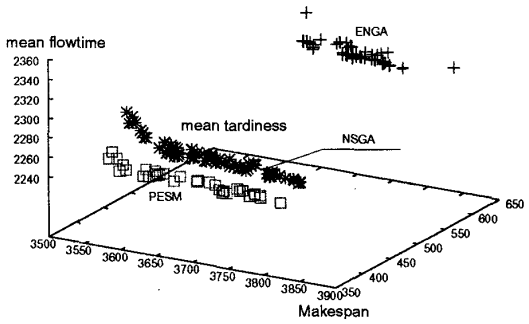



Figure 3: Typical Pareto front for three MOGAs

# 7 Conclusions

The PESM-based genetic algorithm has been applied to solving a multi-objective flowshop problem. Without computing various distance information, the PESM controls the diversity by adjusting selection pressure. Some features of MOGAs, such as ranking and elitism are also adopted in this algorithm.

A measure called outperformance rate has also been proposed. Together with the evaluation of diversity, we have an explicit way to compare two MOGAs for multi-objective problems.

Simulation results show that the PESM proposed in this paper outperforms a fine tuned NSGA and ENGA presented in [6].


## Acknowledgment

This work is partly supported by the Grant-in-Aid for Scientific Researches of the Ministry of Education, Science and Culture of Japan under Grant: B-11450154.

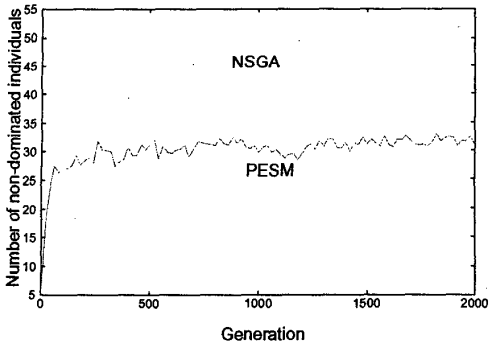



Figure 4: Number of non-dominated individuals for NSGA and PESM

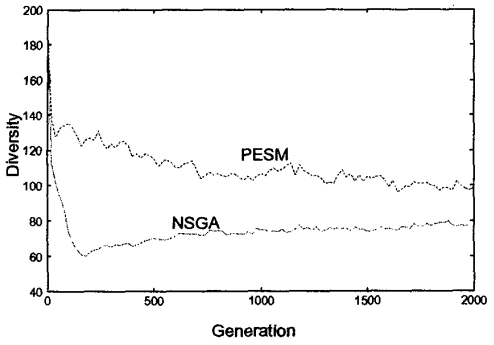



Figure 5: Diversity of populations for NSGA and PESM

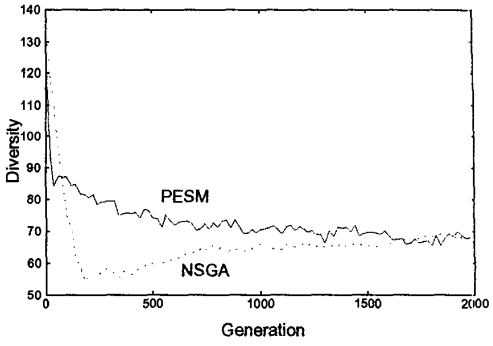



Figure 6: Diversity among non-dominated individuals for NSGA and PESM

# References


[1] N.Srinivas and K.Deb, "Multi-objective function optimization using non-dominated sorting genetic algorithm", Evolutionary Computation, Vol. 2, No. 3, pp. 221-248, 1995.

[2] H.Isibuchi and T.Murata, "Multi-objective genetic local search algorithm", Proceeding of the 1996 International Conference on Evolutionary Computation, pp. 119-224, 1996.

[3] M.R.Garey and D.S.Johnson, *Computers and Intractability*,W.H.Freeman and Company, 1996.

[4] D.Goldberg and L.Wang, "Adaptive nichinng via coevolutionary sharing", Genetic Algorithm in Engineering and Computer Science, John Wiley & Sons Ltd., 1997.

[5] C.Brizuela and N.Sannomiya, "A diversity study in genetic algorithms for jobshop scheduling problems", Proceedings of Genetic and Evolutionary Computation Conference, Vol. 1, pp. 75-82, 1999.

[6] T.P.Bagchi. *Multiobjective Scheduling by Genetic Algorithms*, Kluwer Academic Publisher, 1999.

[7] M.Gen and R.Cheng, *Genetic Algorithm & Engineering Optimization*, A Wiley-Interscience Publication, 2000.

[8] C.Brizuela, N.Sannomiya and Y.Zhao, "Multi-objective flow-shop: preliminary results", in [10], pp. 443-458, 2001.

[9] C.C.Coello, "A short tutorial on evolutionary multi-objective optimization", in [10], pp. 21-35, 2001.

[10] E.Zitzler, K.Deb, L.Thiele, C.C.Coello and D.Corne (Eds.), *Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science No. 1993*, Springer-Verlag, 2001.

[11] K.Deb and T.Goel, "Controlled elitist non-dominated sorting genetic algorithms for better convergence", in [10], pp. 67-81, 2001.