Multi-Objective Evolutionary Optimization of Flexible Manufacturing Systems

Jian-Hung Chen
Dept. of Information Engineering
Feng Chia University
Taichung, Taiwan 407, R.O.C.
jh.chen@ieee.org

Shinn-Ying Ho
Dept. of Information Engineering
Feng Chia University
Taichung, Taiwan 407, R.O.C.
syho@fcu.edu.tw

Abstract

This paper describes multi-objective evolutionary optimization of process planning in flexible manufacturing systems (FMSs). FMS can be described as an integrated manufacturing system consisting of machines, computers, robots, and automated guided vehicles (AGVs). While FMSs give great advantages through the flexibility, FMSs pose complex problems on multi-objective process planning. An evolutionary approach using a multi-objective evolutionary algorithm with a new elite clearing mechanism is proposed for solving the multi-objective process planning problems (MOPPPs). The experimental results demonstrate that our algorithm can solve MOPPPs efficiently.

1 INTRODUCTION

A flexible manufacturing system (FMS) is a production system consisting of a set of identical and/or complementary numerically controlled machines which are connected through an automated guided vehicle (AGV) system. Since FMS is capable of producing a variety of part types and handling flexible routing of parts instead of running parts in a straight line through machines, FMS gives great advantages through the flexibility, such as dealing with machine and tool breakdowns, changes in schedule, product mix, and alternative routes. Flexible manufacturing is of increasing importance in advancing factory automation that keeps a manufacturer in a competitive edge.

While FMS offers many strategic and operational benefits over conventional manufacturing systems, its efficient management requires solutions to complex process planning problems with multiple objectives and constraints. The aim of process planning is to develop a cost effective and operative process plan over the planning phases. Decisions regarding the process planning problem have to be made before the start of actual production, and consists of organizing the limited production resource constraints efficiently. Generally, the process planning includes routing optimization, equipment optimization and machine optimization (Tempelmeier and Kuhn, 1993).

During the past decades, a number of computer-aided process planning (CAPP) systems have been developed for the automated planning and increased efficiency of process planning, considering only a single objective. However, from a system designer’s point of view, it is very desirable to obtain optimal solutions considering all the objectives. Moreover, obtaining a set of non-dominated solutions provides the flexibility for reconfigureable manufacturing.

Recently, some authors applied genetic algorithms (GAs) (Goldberg, 1989) to the process planning. Awadh et al. proposed a CAPP model based on GAs (Awadh, Sepehri and Hawaleshka, 1995). Moon et al. proposed an evolutionary algorithm for solving the flexible process sequencing problems with two objectives (Moon, Li and Gen, 1998). Brandimarte proposed a two-objective hierarchical approach based on a decomposition into a machine loading and a scheduling sub-problem (Brandimarte, 1999). However, the simpified/bicriteria model considered does not satisify the needs of FMSs.

Considering the practical manufacturing environments, we formulate the problems of process planning in FMSs as multi-objective process planning problems (MOPPPs), and propose an evolutionary approach using multi-objective evolutionary algorithm with a new elite clearing mechanism for solving MOPPPs.

This paper is organized as follows: Section 2 introduces the flexible manufacturing system and the mathematical formulation of MOPPPs. Section 3 presents the multi-objective evolutionary algorithm for the problem. Section 4 presents the experimental analysis of the proposed algorithms, and Section 5 summarizes our conclusions.

2 FLEXIBLE MANUFACTURING SYSTEM

A brief description of the flexible manufacturing environment and the mathematical formulation of MOPPPs are given in this section.

2.1 THE FLEXIBLE MANUFACTURING ENVIRONMENT

An FMS consists of a set of identical and/or complementary numerically controlled machines and possibly tool storage. All components are connected through an AGV system. Figure 1 from (Tempelmeier and Kuhn, 1993) shows the layout of a simple FMS with several machines and a tool system.

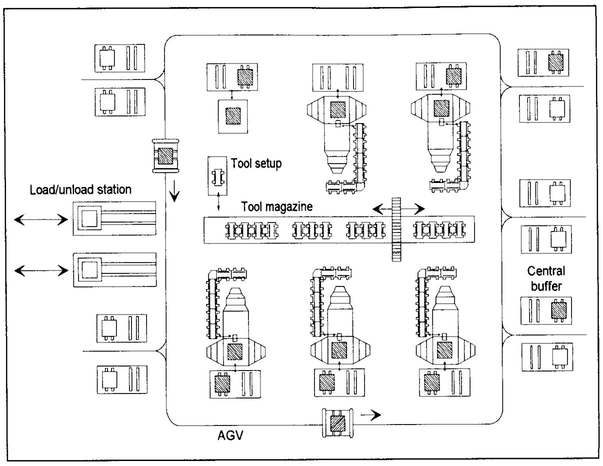

Figure 1: FMS with two AGVs and a central tool magazine.

In order to tackle the process planning in FMSs, the environment within which the FMS under consideration operates can be described as follows:

(1) A part type requires a number of operations. There is a number of part types will be manufactured simultaneously in batches. Parts can choose one or more machines at each of their operation stages, and the transportation of the parts within different machines is handled by an AGV system.

(2) The types and number of machines are known. There is sufficient input/output buffer space at each machine.

(3) A machine type can perform several types of operations, and an operation can be performed on alternative machine types.

(4) A machine type can only process an operation at one time. Operations to be performed in the machine type are nonpreemptive. Operation lot splitting is ignored.

(5) The tool costs of operations in machine types are known. Processing, times of operations in machine types are available and are deterministic.

(6) Workload on each machine is contributed by those operations assigned to a machine.

(7) A load/unload (L/U) station serves as a distribution center for parts not yet processed and as a collection center for parts finished. All vehicles start from the L/U station initially and return to there after accomplishing all their assignments. There is sufficient input/output buffer space at the (L/U) station.

(8) Number of AGVs is given and the transportation time of AGVs are known. AGVs carry a limited number of products at a time. They move along predetermined paths, with the assumption of no delay because of congestion. Preemption of trips is not allowed.

(9) It is assumed that all the design, layout and set-up issues within FMS have already been resolved.

(10) Real-time issues, such as traffic control, congestion, machine failure or downtime, scraps, rework, and vehicle dispatches for battery changer are ignored here and left as issues to be considered during real-time control.

2.2 MATHEMATICAL FORMULATION

Multi-objective process planning problems (MOPPPs) are concerned with the selection of individual process plans for all the parts with minimizing the total flow time, balancing the machine workload, minimizing the machine workload and minimizing the total equipment cost. MOPPPs can be formulated as follows.

2.2.1 Notations

In order to formulate MOPPPs, the following notations are introduced:

i : part index (i = 1, 2, 3, …, np)

j : operation index for part type i (j = 1, 2, 3, …, no_i)

k, l : machine index (k, l = 1, 2, 3, …, nt)

pv_i : production volume (unit) for part type i

pt_{ijk} : processing time per unit to perform operation j of part type i using machine type k

tw_k : workload in machine k, $tw_k = pt_{ijk} * pv_i$

ew : average workload of machines

$$s_{ikl}:\begin{cases}1, \text{ if part type } i \text{ is to transfer from machine } k \text{ to } l\\ 0, \text{ otherwise}\end{cases}$$

$$x_{ijk}:\begin{cases}1, \text{ if machine type } k \text{ selected to perform}\\ \quad \text{operation } j \text{ of part type i}\\ 0, \text{ otherwise}\end{cases}$$

abl : available capacity of AGV per trip

n_{ikl} : the number of trips between machine types k and l for part type i, $n_{ikl} = s_{ikl} \times \left\lceil \frac{pv_i}{abl} \right\rceil$

tm_{kl} : transportation time from machine k to l

t_{ikl} : total transportation time between machines k and l for part type i, $t_{ikl} = n_{ikl} \times tm_{kl}$

c_{ijk} : tool costs to perform operation j of part type i using machine type k

2.2.2 Objectives

There are four objectives to be optimized in FMSs according to the suggestion of (Tempelmeier and Kuhn, 1993), described as follows.

(1) Minimization of the total flow time. This objective is to minimize the processing time and transportation time for producing the parts. The total machine processing time (f_1) is defined as Equation (1), the transportation time (f_2) is defined as Equation (2), and the total flow time (F_1) is defined as Equation (3). Transportations between unlinked machines are penalized in f_2.

$$f_1 = \sum_{i}^{np} \sum_{j}^{no_i} \sum_{k}^{nt} pv_i \cdot pt_{ijk} \cdot x_{ijk} \qquad (1)$$

$$f_2 = \sum_{i}^{np} \sum_{j}^{no_i-1} \sum_{k}^{nt} \sum_{l}^{nt} t_{ikl} \cdot x_{ijk} \cdot x_{i(j+1)l} \qquad (2)$$

$$F_1 = f_1 + f_2 \qquad (3)$$

(2) Minimization of the deviations of machine workload. Balancing the machine workload can avoid creating bottleneck machines. The objective function (F_2) is defined as Equation (4).

$$F_2 = \sum_{k}^{nt} (tw_k - ew)^2 \qquad (4)$$

(3) Minimization of the greatest machine workload. Pursuing this objective also implies attempting to minimize the total flow time. The objective function (F_3) is defined as Equation (5).

$$F_3 = \max (tw_k) \qquad (5)$$

(4) Minimization of the tool costs. Tool costs consider the consumptions of tools. Owing to some unique tools are expensive, it is necessary to consider the tool life. The objective function (F_4) is defined as Equation (6).

$$F_4 = \sum_{i}^{np} \sum_{j}^{no_i} \sum_{k}^{nt} c_{ijk} \cdot x_{ijk} \qquad (6)$$

2.2.3 Multi-objective Mathematical Model

The overall multi-objective mathematical model of MOPPPs can be formulated as follows.

minimize F_1, F_2, F_3, F_4

subject to

$$\sum_{k}^{nt} x_{ijk} = 1, \forall (i, j) \qquad (7)$$

The operation flexibility is concerned with an operation can be performed on alternative machines with the different processing time and transportation time. The constraint, Equation (7), ensures that only one machine type is selected for each operation of a part type.

3 MULTI-OBJECTIVE EVOLUTIONARY APPROACH

Multi-objective evolutionary algorithms have been recognized to be particularly suitable for solving MOOPs because the ability to exploit and explore multiple solutions in parallel, and the ability to find an entire set of Pareto-optimal solutions in a single run.

We applied and refined the generalized multi-objective evolutionary algorithm (GMOEA) proposed by us (Ho and Chang, 1999), and propose a new clearing elite mechanism to reduce the non-dominated set. The advantages of GMOEA are:

(1) Elitism: GMOEA incorporates with two populations: the current population and the elite population, called the tentative set of non-dominated solutions (TSONS).

(2) Fitness assignment strategy: The generalized Pareto-based scale-independent (GPSI) fitness function can assign discriminative fitness value to individuals.

(3) Intelligent crossover (IC): IC is introduced to improve the performance of GMOEA on solving problems with a large number of parameters.

The representation of the chromosome is presented in Section 3.1. The fitness assignment strategy and IC are described in Sections 3.2 and 3.3, respectively. The mutation operation approach is described in Section 3.4. Section 3.5 presents the new elite clearing mechanism. The flow of our algorithm is provided in Section 3.6.

3.1 CHROMOSOME REPRESENTATION

The chromosome representation is defined a series of the operations for all the parts. In the chromosome, each gene stands for a machine type number for the machining operations. The assignment of machine types to operations is made by generating random numbers within the range [1, *nt*], so that the corresponding machine numbers are determined. Take Figure 2 for example, [4 2 3 … *nt*] stands for the process plan of part 1, [3 4 *nt* 2 3 … 1] stands for the process plan of part 2, [4 5 1 …] stands for the process plan of part *n*p.

Part *1*	Part *2*	…	Part *np*
4 2 3… *nt*	3 4 *nt* 2 3 … 1	…	4 5 1 …

Figure 2: The representation of a chromosome.

3.2 FITNESS ASSIGNMENT

The fitness assignment strategy of GMOEA uses a generalized Pareto-based scale-independent (GPSI) fitness function considering the quantitative fitness values in Pareto space for both dominated and non-dominated individuals.

Let GPSI fitness function be a tournament-like score for an individual x_u at the l^{th} evaluation operation with corresponding objective vector u. The current position of x_u in the individuals' score can be given by

$$\text{score}\ (x_u, l) = p_u^l - q_u^l + C \qquad \textbf{(8)}$$

where p_u^l is the number of individuals which can be dominated by x_u and q_u^l is the number of individuals which can dominate x_u in the current Pareto space. The constant C is used to obtain the positive fitness value (generally assign the number of the participant individuals).

3.3 INTELLIGENT CROSSOVER

Two parents breed two children using IC at a time by means of orthogonal array (OA). OA is an array of numbers whose columns are pairwise orthogonal. In every pair of columns all ordered pairs of numbers occur the same number of times.

An OA used in IC is described as follows. Let there be α factors, with two levels (or treatments) for each factor. The total number of experiments is 2^{α} for the popular "one-factor-at-a-time" study. The columns of two factors are orthogonal when the four pairs, (1,1), (1,2), (2,1), and (2,2), occur equally frequently over all experiments. When any two factors in an experimental set are orthogonal, the set is called an OA. To establish an OA of α factors with two levels, we obtain an integer $\beta = 2^{\lceil \log_2(\alpha+1) \rceil}$, build an orthogonal array $L_\beta(2^{\beta-1})$ with β rows and (β-*1*) columns, and use the first α columns. For instance, Table I shows an orthogonal array L_8 (2^7). Orthogonal experiment design can reduce the number of experiments for factor analysis. Generally, levels 1 and 2 of a factor represent selected genes from parents 1 and 2, respectively.

Table 1: Orthogonal array $L_8(2^7)$

Exp. no.	Factors 1	2	3	4	5	6	7	Function Evaluation value
1	1	1	1	1	1	1	1	y_1
2	1	1	1	2	2	2	2	y_2
3	1	2	2	1	1	2	2	y_3
4	1	2	2	2	2	1	1	y_4
5	2	1	2	1	2	1	2	y_5
6	2	1	2	2	1	2	1	y_6
7	2	2	1	1	2	2	1	y_7
8	2	2	1	2	1	1	2	y_8

Let y_t be the positive function evaluation value of experiment no. *t*. Define the main effect of factor *j* with level k , S_{jk},

$$S_{jk} = \sum_{t=1}^{\beta} Y_t^2 \times [\text{the level of experiment number t of factor j is k}],$$

where

$$[condition] = \begin{cases} 1 & if \quad the \quad condition \quad is \quad true \\ 0 & otherwise\ , \end{cases}$$

and $$Y_t = \begin{cases} y_t & \text{if the function is to be maximized} \\ 1/y_t & \text{if the function is to be minimized} \end{cases}$$

The steps to use the OA to achieve the IC is described as follows:

Step 1: Select the first α columns of OA $L_\beta(2^{\beta-1})$ where $\beta = 2^{\lceil \log_2(\alpha+1) \rceil}$. Note that let the chromosome be uniformly separated into α sub-strings and each sub-string of a chromosome be regarded as a factor in OA.

Step 2: Let level 1 and level 2 of factor j represent the j^{th} sub-string of a chromosome coming from the parent 1 and parent 2, respectively. Generate by-product individuals by means of OA.

Step 3: Calculate the values of the *l* objectives for each solution in the β by-product individuals. Compute their fitness value y_t for experiment no. *t* where *t* = 1, 2, …, β, and then update TSONS.

Step 4: Compute the main effect S_{jk} where *j = 1, 2, …,* α and *k = 1, 2*.

Step 5: Determine the best level for each sub-string. Select level 1 for the j^{th} sub-string if $S_{j1} > S_{j2}$.

Otherwise, select level 2.

Step 6: The chromosome of the first child is formed from the best combinations of the better sub-string from the derived corresponding parents.

Step 7: Rank the most effective factors from rank 1 to rank α. The factor with large (MED) has higher rank.

Step 8: The chromosome of the second child is formed similarly as the first child except that the sub-string with the lowest rank adopts the other level.

Since the machine index can be duplicated in a chromosome, no infeasible solutions will be generated when applying IC.

3.4 MUTATION

The procedure of mutation operator is as follows:

Step 1: Randomly select a machine index in the chromosome. Let the machine index be *i*.

Step 2: Replace the machine index *i* by randomly generate an integer value from the range [1, *nt*].

3.5 THE ELITE CLEARING MECHANISM

The main idea of the elite population is to improve the performance of the algorithm. Therefore, the individuals in the elite population may influences the behavior of the algorithm. However, the elite population may biased towards certain regions of the search space, leading to the unbalanced search directions of the algorithm. Thus, pruning the elite population to encourage the search directions toward unexplored regions is necessary.

Based on the idea of encouraging the search toward unexplored regions, the current population is used to represent the explored regions. If a non-dominated individual covers more individuals in the current population, it implies that the covered region is well explored. Therefore, once the size of the elite population exceeds the upperbound, these non-dominated individuals can be cleared from the elite population. By the way, it is also necessary to keep the boundary individuals, because the boundary individuals in each objective are the representative points for guiding the search directions. The procedure of elite clearing mechanism is as follows:

Step 1: For each non-dominated individual *i* in the elite population, calculate the number of individuals it dominates, *di*, in the current population. Let *di* of the boundary individuals in each objective be –1, so that they are always survived.

Step 2: Select a individual with larger d_i to be cleared by binary tournament selection. Clear a number of non-dominated individuals until the elite population achieves the upperbound.

3.6 MULTI-OBJECTIVE EVOLUTIONARY ALGORITHM

The flow of GMOEA is as follows:

Step 1: Initialization: Randomly generates an initial population of N_{pop} solutions. Let the initial TSONS be empty.

Step 2: Evaluation: Calculate the values of the *l* objectives for each solution in the current population and then compute the GPSI fitness function as the fitness values of all individuals.

Step 3: Update TSONS: Copy the non-dominated individuals and remove the dominated individuals in TSONS. Reduce the number of individuals by means of the elite clearing mechanism.

Step 4: Selection: Select ($N_{pop} - N_{ps}$) individuals from the population by binary tournament selection, and select N_{ps} non-dominated solutions from the TSONS randomly to form the new population, where N_{ps} is equal to $N_{pop} * P_s$.

Step 5: Crossover: Select ($N_{pop} * p_c$) parents for crossover operations. Apply IC for all the selected pairs of parents.

Step 6: Mutation: Apply the mutation operator.

Step 7: Termination test: If the termination conditions are satisfied, end the algorithm. Otherwise, return to Step 2.

4 EXPERIMENT RESULTS

In order to investigate the performance of GMOEA, GMOEA is tested with the multi-objective 0/1 knapsack problems (Zitzler and Thiele, 1999). SPEA is also implemented to solve MOPPPs in order to make a direct comparison. The performance measure of algorithms we used is the coverage ratio of two set (A, B) by (Zitzler and Thiele, 1999). The coverage ratio of set (A, B) is calculated as follows:

$$C(A,B) := \frac{\text{the number of individuals of B dominated by A}}{\text{the number of individuls of B}}$$

The value C(A, B) = 1 means that all individuals in B are dominated by A. The opposite, C(A, B)=0, denotes that none of individuals in B are dominated by A.

4.1 COMPARSIONS OF MULTI-OBJECTIVE KNAPSACK PROBLEMS

The parameter settings of GMOEA for solving the multi-objective 0/1 knapsack problem with 750 items are as follows:

Current population size	:	50
Upperbound size of TSONS	:	50
Selection rate (Ps)	:	0.25
Crossover rate (P_c)	:	0.8

Mutation rate (P_m) : 0.01

Columns of OA (α) : 15

30 independent runs were performed per test problems, compared with same function evaluation times of SPEA. The raw results of SPEA are from the author's website. The experimental result of 2 knapsack-750 items is shown in Figure 3. The results concerning the C measure are shown in Table 2.

Generally, the simulation results of knapsack problems prove that GMOEA do better than SPEA. While SPEA use a large number of population size (250,300,350), none of solutions found by GMOEA are dominated by the solutions of SPEA.

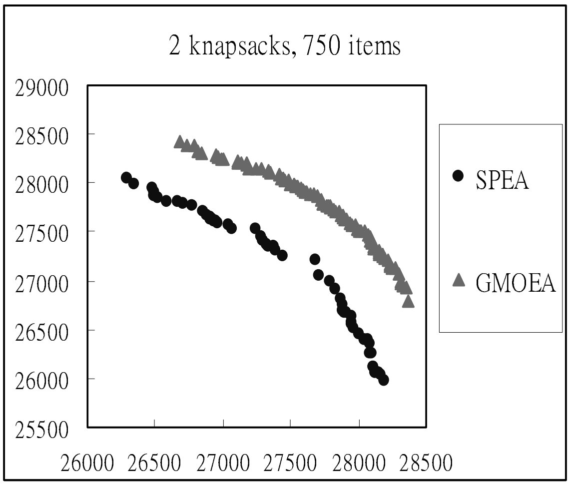

Figure 3: Trade-off fronts out from 30 runs.

Table 2: The C measure of GMOEA and SPEA.

Knapsacks problems	**2-750**	**3-750**	**4-750**
Number of solutions found by SPEA	37	426	1751
Number of solutions found by GMOEA	94	301	372
C(GMOEA, SPEA)	1 (37/37)	0.57 (244/426)	0.72 (1261/1751)
C(SPEA, GMOEA)	0 (0/94)	0 (0/301)	0 (0/372)

4.2 COMPARSION OF MOPPPS

Since MOPPPs are related to the generalized assignment problem (GAP) (Tempelmeier and Kuhn, 1993) (Barndimarte, 1999). Therefore, we used the benchmark problem instances of GAP, which are provided by OR-Library. Two instances, (20 agents, 100 jobs) and (20 agents, 200 jobs) in the benchmark – gapd are derived and formulated. Let agents be machines, jobs be operations, the cost of allocating job to agent be the processing time pt_{ijk}, and the resource requirement be the tool costs c_{ijk} in FMS. Assume a part is consists of 5 operations, so that the first instance has 20 parts, the second instance has 40 parts. The production volume (PV_i) of each part types is given as follows: {45, 43, 39, 46, 42, 56, 37, 33, 61, 30, 55, 43, 24, 39, 29, 44, 30, 45, 29, 30, 55, 33, 37, 43, 62, 36, 42, 44, 53, 40, 35, 41, 34, 29, 38, 49, 43, 25, 69, 41}, i = 0, 1, … , 40. Let the available capacity of AGV, *abl*, be 10. Considering the real manufacturing environment, the transportation time of AGV is given in Table 4. The transportation time within the same machine is to reflect that a machine unit may be a combination of several machines.

The parameter settings of GMOEA are as follows.

Current population size : 50

Upperbound size of TSONS : 50

Selection rate (Ps) : 0.2

Crossover rate (P_c) : 0.6

Mutation rate (P_m) : 0.05

Columns of OA (α) : 15

The parameter settings of SPEA are the same as the settings of GMOEA, except the population size of SPEA is 150 and the elite population is 50. 30 independent runs were performed per test problems, compared with function evaluation times = 100000.

Table 4: The C measure of GMOEA and SPEA.

MOPPPs	**20 machines 100 operations**	**20 machines 200 operations**
Number of solutions found by SPEA	415	199
Number of solutions found by GMOEA-N	392	250
Number of solutions found by GMOEA	465	313
C(GMOEA-N, SPEA)	0.71 (295/415)	0.90 (180/199)
C(SPEA, GMOEA-N)	0 (0/392)	0 (0/250)
C(GMOEA, SPEA)	1 (414/415)	1 (199/199)
C(SPEA, GMOEA)	0 (0/465)	0 (0/313)

In order to investigate the affects of the elite clearing mechanism, GMOEA without the elite clearing mechanism (GMOEA-N) is also performed. Moreover, box plots are used to visualize the distribution of solutions in each objective.

Box plots of MOPPP with 20 machines and 200 operations are shown as Figure 4, 5, 6 and 7. The results concerning the C measure are shown in Table 4. The simulation results of MOPPPs indicate that all the non-dominated solutions found by SPEA are dominated by GMOEA, and the elite clearing mechanism improves the distribution of solutions while maintaining the quality of solutions.

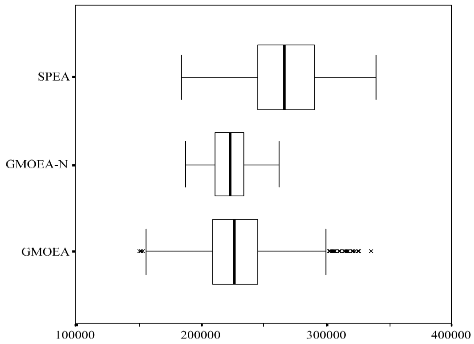

Figure 4: The distribution of solutions in F_1.

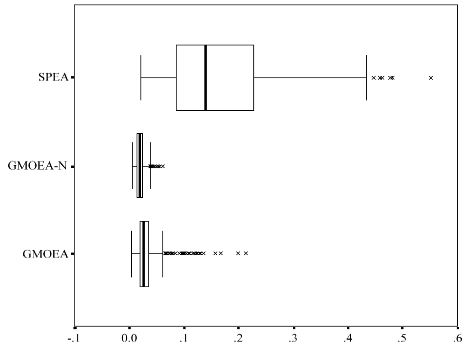

Figure 5: The distribution of solutions in F_2.

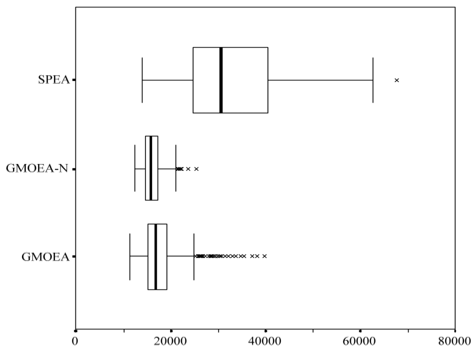

Figure 6: The distribution of solutions in F_3.

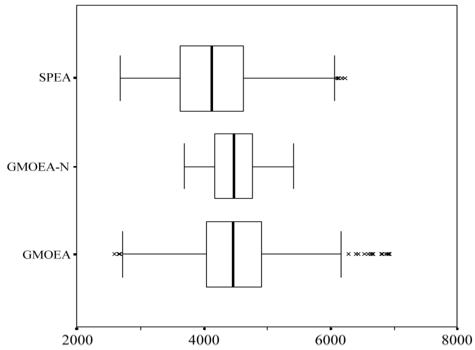

Figure 7: The distribution of solutions in F_4.

4.3 DISCUSSIONS

From the reported results, it is shown that:

(1) The quality of non-dominated solutions obtained by GMOEA is superior to that of SPEA, and GMOEA outperforms SPEA in convergence speed and high accuracy within the same function evaluation times.

(2) GMOEA uses a compact population while SPEA uses a larger number of population, and no sharing or clustering technique is used in GMOEA. Therefore, the actual computation time of GMOEA is lesser than SPEA, because the complexity of identifying the non-dominated solutions is $O(N^2)$.

(3) From the experimental results of GMOEA and GMOEA-N. It is shown that the elite clearing mechanism is able to encourage the algorithms to explore the unexplored search regions, so that the distribution of solutions can be improved. Moreover, the elite clearing mechanism is simple and efficient than the clustering technique used in SPEA.

5 CONCLUSIONS

Multi-objective process planning problem (MOPPP) is an important problem in the pre-release planning phase of flexible manufacturing systems. This paper has presented an evolutionary approach using multi-objective evolutionary algorithm with a new elite clearing mechanism for solving MOPPPs. Objectives considering the flow time, machine balancing, machine workload and tool cost are optimized simultaneously. Experimental results demonstrated the proposed approach is suitable to solve the complex industrial problems with a large number of parameters.

References

S.-Y. Ho and X.-I. Chang (1999). An efficient generalized multiobjective evolutionary algorithm *GECCO-99: Proceeding of Genetic and Evolutionary Computation Conference*, pp. 871-878.

H. Tempelmeier and H. Kuhn (1993). *Flexible manufacturing systems: decision support for design and operation*. John Wiley & Sons.

B. Awadh, N. Sepehri and O. Hwalwshka (1995) A computer-aided process planning model based on genetic algorithms. *Computers and Operations Research*, 22(8): 841-856.

C. Moon, Y.-Z. Li and M. Gen (1998). Evolutionary algorithm for flexible process sequencing with multiple objectives. *Proceeding of IEEE International Conference on Computational Intelligence*, pp. 27-32.

E. Zitzler and L. Thiele (1999). Multiobjective evolutionary algorithms: a comparative case study and the strengthen Pareto approach. *IEEE Transaction on Evolutionary Computation*, **3**(4): 257-271.

P. Brandimarte (1999). Exploiting process plan flexibility in production scheduling: a multi-objective approach. *European Journal of Operational Research* (114): 59-71.

D. E Goldberg (1989). *Genetic Algorithms in search, Optimization and Machine Learning*, Addison – Wesley Publishing Company.

G. Taguchi and S. Konishi (1987). *Orthogonal Arrays and Linear Graphs. Dearbon, MI: American Supplier Institute*.

Table 3: Transportation time of AGV from machine to machine.

-: Represent there are no routing path between machines.

From\To	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	4	11	17	8	15	16	14	-	6	8	13	12	16	6	-	8	17	16	8	14
2	6	3	15	9	17	8	7	16	3	6	18	-	3	6	17	-	13	11	-	8
3	6	18	5	7	13	11	15	5	17	14	17	13	11	12	16	13	6	9	10	12
4	17	15	16	3	5	4	-	13	13	4	3	3	7	17	16	6	12	11	4	6
5	13	15	4	11	4	7	4	18	15	8	10	-	17	11	12	18	17	8	9	16
6	11	8	15	8	13	5	17	12	13	16	5	14	11	16	17	16	15	-	9	18
7	4	-	18	9	13	5	3	5	3	9	10	18	15	12	6	7	3	-	-	11
8	9	16	18	16	9	14	8	3	-	18	13	11	16	3	-	6	16	11	-	3
9	12	-	5	7	12	17	5	11	4	-	7	18	7	17	11	4	11	9	15	9
10	-	9	3	11	9	-	-	3	7	3	7	13	18	3	15	10	17	6	16	9
11	9	17	13	12	-	5	8	10	-	18	4	14	-	15	14	-	8	15	10	4
12	4	14	6	15	3	17	3	4	3	7	6	3	-	12	-	15	18	12	18	4
13	-	4	12	7	16	10	4	17	17	18	12	-	3	18	4	3	8	15	11	-
14	13	4	15	-	12	4	15	15	5	8	9	8	4	3	15	17	8	-	8	4
15	4	-	17	18	4	10	-	18	16	10	18	16	9	12	4	10	13	8	12	18
16	3	4	18	10	6	4	3	11	7	9	-	15	12	17	9	3	4	11	6	11
17	-	11	17	15	6	5	4	-	8	12	10	9	16	3	-	18	4	8	-	5
18	15	6	9	6	14	6	17	14	5	-	9	10	17	3	3	3	5	3	12	6
19	3	10	9	10	16	-	15	18	-	4	13	9	-	18	11	5	3	12	4	13
20	5	15	-	18	16	17	12	13	-	17	-	16	5	-	16	-	18	5	16	3