APEM journal

Advances in Production Engineering & Management
Volume 9 | Number 3 | September 2014 | pp 139–151
http://dx.doi.org/10.14743/apem2014.3.183

ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

Scheduling optimization of a flexible manufacturing system using a modified NSGA-II algorithm

Nidhiry, N.M.[a,*], **Saravanan, R.**[b]

[a]Department of Mechanical Engineering, Karapagam University, Coimbatore, Tamil Nadu, India
[b]Department of Mechanical Engineering, Sri Krishna College of Technology, Coimbatore, Tamil Nadu, India

A B S T R A C T

The Flexible Manufacturing System (FMS) belongs to the class of production systems in which the main characteristic is the simultaneous execution of several processes and sharing a finite set of resources. Nowadays, FMS must attend to the demands of market needs for personalized products. Consequently the life-cycle of a product tends to be shorter and a greater variety of products must be produced in a simultaneous manner. The FMS considered in this work has 16 CNC machine tools for processing 80 varieties of products. Since the minimizing of a machine's idle time and thus the minimizing of total penalty costs are contradictory objectives, the problem has a multi-objective nature. The objective of this research was to develop a modified non-dominated sorting genetic algorithm (NSGA-II) for multi-objective optimization. The research will then evaluate and discuss the performance of the modified NSGA-II against the original NSGA-II. The existing NSGA II has been modified in order to improve the global optimal front and reduce the computational effort. The result has been compared with the existing NSGA-II, cuckoo search (CS), particle swarm optimization algorithm (PSO), etc. and it was found that the proposed approach was superior.

A R T I C L E I N F O

Keywords:
Flexible manufacturing system
Scheduling optimization
Multi-objective optimization
NSGA-II
Modified NSGA-II

**Corresponding author:*
Email: nidhishnidhiry@gmail.com
(Nidhiry, N.M.)

Article history:
Received 20 October 2013
First Revision 16 June 2014
Second Revision 16 August 2014
Accepted 19 August 2014

1. Introduction

FMS operational decisions consist of pre-release and post-release decisions. FMS planning problems also known as pre-release decisions take into account the pre-arrangement of parts and tools before the operation of FMS begins. The problem of scheduling of FMS, which come under the category of post release decisions deal with the sequencing and routing of the parts when the system is in operation. The problem of loading of machine in an FMS is specified so as to assign the machine, operations of selected jobs, and the tools necessary to perform these operations by satisfying the technological constraints (available machine time and tool slots constraint) in order to ensure the unbalance of the system is minimum with maximum throughput, when the system is on operation. An attempt has been made to solve the objective function and simultaneously to bring the outcome in close proximity to the real assumption of the FMS environment. There are a number of problems faced during the life cycle of an FMS. These problems are classified into design, planning, scheduling and control problems. In particular, task of scheduling and the control problem during the operation are important owing to the dynamic nature of the FMS such as flexible parts, tools and routings of automated guided vehicle (AGV). Scheduling of operations is one of the most critical issues in the planning and managing of manufacturing processes. The increased use of flexible manufacturing systems (FMS) that effectively provides a customer with diversified products has created a significant set of operational challenges. The design of these kinds of systems is characterized by massive alternatives of positions and paths of components, while in practice there is always the attempt to minimize the cycle time, dealing with a lot of alternatives in respect to positioning of components and paths' planning.

1.1 Earlier research

During the last three decades much research has been done in this area. Many heuristic algorithms have been developed to generate optimum schedule and part-releasing policies. Most of these algorithms include enumerative procedures, mathematical programming and approximation techniques, i.e. linear programming, integer programming, goal programming, dynamic programming, transportation and network analysis, branch and bound, Lagrangian relaxation, priority-rule-based heuristics, local search algorithms (ITS, threshold algorithm, Tabu search, SA), genetic algorithm (GA), etc. Of these techniques, some are specific to particular objectives, and some are specific to particular instances with respect to time needed for computational.

Guo et al. [1] presented a comprehensive review of genetic algorithm based optimization model for scheduling flexible assembly lines. In this paper a scheduling problem in the flexible assembly line is investigated and a bi-level genetic algorithm to solve the scheduling problem is developed. Tiwari and Vidyarthi [2] proposed a genetic algorithm based heuristic to solve the machine loading problem of a random type FMS. The proposed GA based heuristic determines the part type sequence and the operation machine allocation that guarantee the optimal solution to the problem. In another scheduling paper [3], taking into account only 6 machines and 6 jobs. Kumar, Tiwari and Shankar [4], analyzed ant colony optimization approach (ACO) in FMS scheduling. But ACO algorithm performs better in problem such as traveling sales, the vehicle rooting etc. In previous years most research concerning the AGV scheduling has been focused on development of scheduling algorithms for a single objective such as minimizing of setup cost or minimizing the loading and unloading time. Toker, Kondakci and Erkíp [5] proposed an approximation algorithm for the n jobs and m machines resource constraint job shop problem. Hoitomt et al. [6] explored the use of the Lagrangian relaxation technique to schedule job shops characterized by multiple non-identical machine types, generic procedure constraints and simple routing considerations. He and Kusiak [7] addressed three different industrial scheduling problems, with heuristic algorithms for each problem. Lee and DiCesare [8] used Petri nets to model the scheduling problems in FMS. Shnits and Sinreich [9] present the development of a multi-criteria control methodology for FMSs. The control methodology is based on a two-tier decision making mechanism. The first tier is designed to select a dominant decision criterion and a relevant scheduling rule set using a rule-based algorithm. In the second tier, using a look-ahead multi-pass simulation, a scheduling rule that best advances the selected criterion is determined. Yu and Greene [10] use a simulation study to examine the effects of machine selection rules and scheduling rules for a flexible multi-stage pull system. Jerald et al. [11] proposed a combined objective scheduling optimization solution for FMS. Saravanan and Noorul had modified the same problem in scatter-search approach of flexible manufacturing systems, but this work is only for 43 parts and few generations. Sankar et al. [12] applied multi-objective genetic algorithm FMS for 16 machines and 43 jobs. The results were better than conventional optimization approaches. Burnwal and Deb [13] took the same problem and improved results using cuckoo search (CS) based approach. Udhayakumar and Kumanan [14] have generated an active schedules and optimal sequence of job and tool that can meet minimum make span schedule for the flexible manufacturing system. Kumar et al. [15] proposed a machine selection heuristic and a vehicle assignment heuristic which are incorporated in the differential evolution approach to assign the tasks, to appropriate machine and vehicle, and to minimize cycle time. There are also many other interesting approaches regarding simulation in FMS [16-19] as-well-as several heuristic and other algorithms [20, 21] which can be used for multi-objective problem solving in real production environment. Many authors have been trying to emphasize the utility and advantages of genetic algorithm, simulated annealing, and other heuristics.

In this work, modified approach has been proposed based on the non-dominated sorting genetic algorithm-II (NSGA-II) for multi-objective optimization of a specific manufacturing environment with two objectives [22, 23]. The procedures are applied to relatively large-size problems of up to 80 part varieties passing through 16 different CNC machine centres, and the results are found to be closer to the global optimum sequence.

1.2 The main contribution of the paper

The following are the novel aspects in this paper:

- Two new objective functions are considered separately for minimizing penalty cost and minimizing machine idle time. So the optimization model used in this paper is truly an improved one.
- No literature had considered 80 varieties of products for a particular combination of tools in the tool magazines using 16 machines in 5 flexible manufacturing cells (FMC) minimizing penalty cost and minimizing machine idle time. From the results it is proved that the new approach gave better results when compared to other algorithms.
- This paper has considered the advantages of the evolutionary algorithms MOGA and NSGA-II, and developed a modified NSGA-II algorithm for solving the problem.
- Two normalized functions of weighing objective and average fitness factor are used to select the best optimal solution. They are used only for selecting the best Pareto solution from the non-dominated solutions of Pareto optimal fronts obtained from the proposed evolutionary algorithms.
- A user friendly and general purpose software package has been developed in this work for modified NSGA-II algorithm using .NET language that can be used to obtain the optimal solution for any similar problems.

Our proposed optimization methods have the following advantages:

- A global Pareto optimal solution is possible.
- They are easy to program and implement efficiently when compared to conventional optimization techniques.
- The proposed approach consumes only 50 % time in comparing with NSGA-II and is superior in terms of objective function.
- Moreover, the procedure developed in this work can be suitably modified to suit any kind of FMS with a large number of components and machines.
- They offer Pareto optimal fronts that offer more number of optimal solutions for the user to choose from.

2. Problem descriptions

The problem environment, assumption and aim of the present work are as follows:

1. The FMS considered in this work has a configuration as shown in Fig. 1. There are five flexible machining cells (FMCs), each with two to six computer numerical machines (CNCs), an independent and a self-sufficient tool magazine, one automatic tool changer (ATC) and one automatic pallet changer (APC). Each cell is supported by one to three dedicated robots for intra-cell movement of materials between operations. There is a loading station from which parts are released in batches for manufacturing in the FMS. There is an unloading station where the finished parts are collected and conveyed to the final storage area. There is one automatic storage and retrieval system (AS/RS) to store the work in progress. The five FMCs are connected by two identical automated guided vehicles (AGVs). These AGVs perform the inter cell movements between the FMCs, the movement of finished product from any of the FMCs to the unloading station and the movement of semi-finished products between the AS/RS and the FMCs.

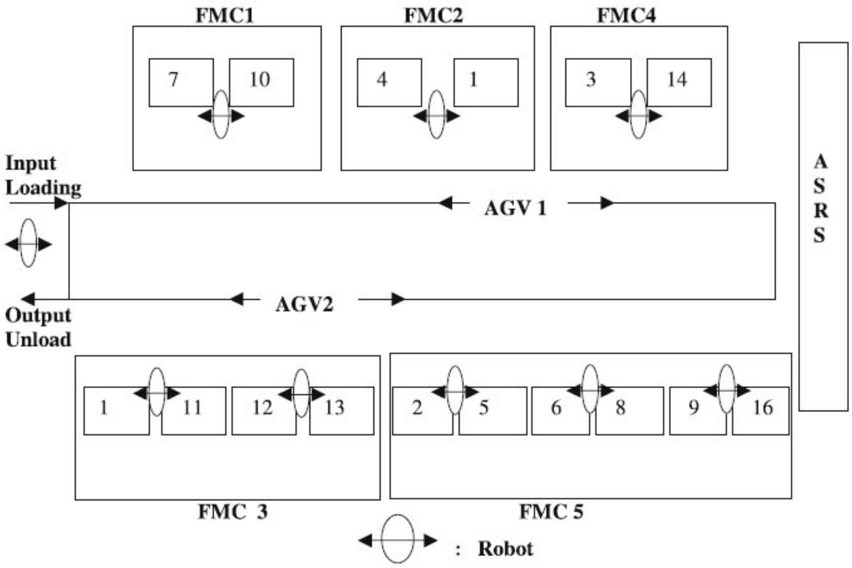

Fig. 1 FMS structure [5]

2. The assumptions made in this work are as follows:
   - There are 80 varieties of products for a particular combination of tools in the tool magazines using 16 machines in 5 FMCs.
   - The type/variety has a particular processing sequence batch size, deadline and penalty cost for not meeting the deadline.
   - Each processing step has a processing time with a specific machine.
   - There is no constraint on the availability of pallets, fixtures, AGVs, robots, automated storage and retrieval system, cutting tools, and part programs as and when they are needed at the required places.
   - A random product-mix generated as shown in the Table 1 reflect the current market demand.
3. The objective of the schedule:
   - Minimizing the machine idle time (TD_i),
   - Minimizing the total penalty cost (TP_i).

$$TD_i = \sum_j MI_j \qquad (j - \text{ machine number}) \tag{1}$$

$$MI_j = TI - \sum_i PT_{ji} \qquad (i - \text{ job number}) \tag{2}$$

$$TP_i = \sum_i (TD_i - DD_i) \times UP_i \times BS_i \tag{3}$$

Nomenclature:

TD_i – Total machine idle time
TI – Total elapsed time
PT_{ji} – Processing time of i-th job on the j-th machine
TP_i – Total penalty cost
PT_i – Processing time of i-th job
DD_i – Due date for i-th job
UP_i – Unit penalty cost for job i
BS_i – Batch size of job i

3. Proposed methodology

As is well-known, a genetic algorithm is a procedure used to find approximate solution to search problems through application of the principles of evolutionary biology. Genetic algorithms uses biologically inspired phenomena such as natural selection, reproduction, crossover and mutation. Genetic algorithms are typically implemented using computer simulations in which an optimization problem is specified.

The two processes together improve an organism's ability to survive with in its environment in the following manner:

- Natural selection determines which organism will have the opportunity to reproduce and survive within a population.
- Reproduction involves genes from two separate individuals combining to form offspring that inherit the survival characteristics of their parents. These algorithms seek to initiate the way in which beneficial gene reproduces themselves through successive population and hence contribute to the gradual ability of an organism to survive.

3.1 NSGA-II algorithm

A multi-objective decision problem is defined as follows. Given an n-dimensional decision variable vector $x = \{x_1,\ldots,x_n\}$ in the solution space X, find a vector x^* that minimizes a given set of K objective functions $z(x^*) = \{z_1(x^*),\ldots,z_K(x^*)\}$. The solution space X is generally restricted by a series of constraints, such as $g_j(x^*) = b_j$ for $j = 1,\ldots,m$, and bounds on the decision variables. Solution to any multi-objective optimization problem is a family of points known as non-dominated solutions or Pareto optimal set, where each objective component of any point along the Pareto-optimal front can only be improved by degrading at least one of its other objective functions. Pareto optimal front is a curve that joins all Pareto optimal set points. If all objective functions of a solution cannot be improved simultaneously, then that solution is said to have non-domination character.

The methodology used to find the optimal solution to this problem is NSGA-II. It is based on a ranking procedure, consisting in extracting the non-dominated solutions for a population and giving them a rank of 1. These solutions are removed from this population; the next group of non-dominated solution has a rank of 2 and so on. The algorithm has a current population that is used to create an auxiliary one (the offspring population); after that, both populations are combined to obtain the new current population. The procedure is as follows: the two populations are sorted according to their rank, and the best solutions are chosen to create the new population. In the case of having to select some individuals with same rank, a density estimation based on measuring the crowding distance to the surrounding individuals belonging to the same rank is used to get the most promising solutions. Typically, both the current and auxiliary population has equal size.

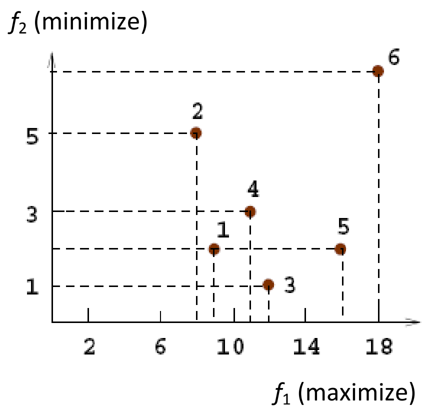

Fig. 2 Values of two objective functions [17]

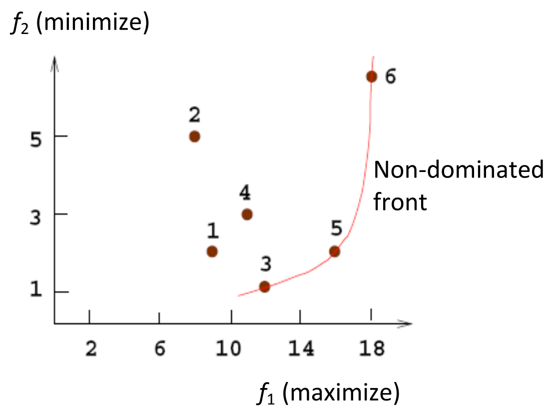

Fig. 3 Pareto-optimal solutions [17]

The concept of dominance is as follows: X_1 dominates X_2 only if X_1 is no worse than X_2 in all objectives, and X_1 is strictly better than X_2 in at least one objective. For example, for two objective functions in Fig. 2, the Pareto-optimal solutions (i.e., non-dominated front) are in Fig. 3. It can be seen that solution 3 dominates solution 2, but it does not dominate solution 5.

3.2 Modified NSGA-II algorithm

The methodology used in this problem is a modified NSGA-II approach to find the optimal solution. The simple GA is modified as a multi objective optimization by including combined objective function (average fitness factor) and non-dominance concept that is used in NSGA-II which is given in the flowchart shown in Fig. 4. After every cycle using combined objective function, new set of solutions is originated. The product sequence obtained after every 500 generations will take and apply NSGA-II algorithm. Then few sequences will be generated with zero dominance count. A new set of optimum solutions will be obtained after 4500 generations.

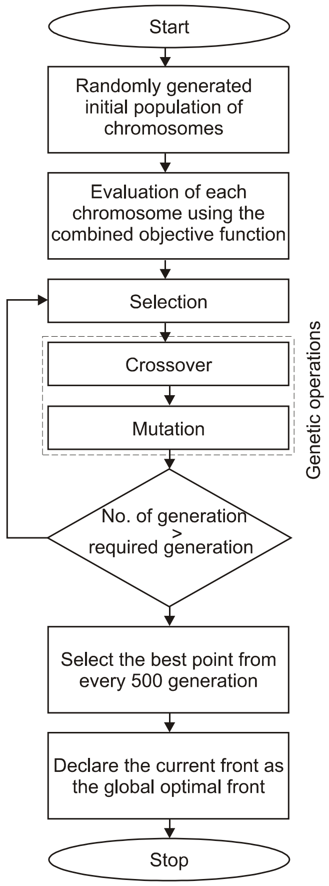

Fig. 4 Flow chart of a modified NSGA-II

3.3 Optimization procedure

Let us suppose the current market demand (Table 1). The objective of the schedule is to minimize the machine idle time (TD_i) and the total penalty cost (TP_i). Combined objective function (COF) is:

$$Objective(1) = \frac{TD_i - Min.TD_i}{Max.TD_i - Min.TD_i} \tag{4}$$

$$Objective(2) = \frac{TP_i - Min.TP_i}{Max.TP_i - Min.TP_i} \tag{5}$$

$$Average\ fitness\ factor = \frac{Objective(1) + Objective(2)}{2} \tag{6}$$

Table 1 Machining sequence, time, deadline, batch size, and penalty details

Part No.	Processing sequence – {Machine No., Processing time (min)}	Deadline (days)	Batch size (Nos)	Penalty cost (INR/unit/day)
1	{6, 1}, {7, 1}, {8, 1}, {10, 2}	17	150	1.00
2	{2, 1}, {6, 1}, {8, 1}, {9, 2}, {14, 4}, {16, 2}	17	200	1.00
3	{8, 1}, {11, 3}, {13, 4}	14	800	1.00
4	{9, 4}	26	700	2.00
5	{4, 5}, {5, 3}, {15, 4}	11	150	1.00
6	{6, 5}, {14, 1}	16	700	1.00
7	{3, 5}, {6, 3}, {16, 5}	26	250	2.00
8	{5, 4}, {6, 5}, {8, 1}	26	850	2.00
9	{4, 1}, {5, 5}, {8, 1}, {11, 1}	1	100	0.00
10	{2, 2}, {9, 1}, {16, 4}	20	150	2.00
11	{8, 4}, {12, 2}	1	250	1.00
12	{6, 2}, {8, 4}, {10, 1}	19	1000	3.00
13	{6, 1}, {7, 5}, {10, 4}	25	700	4.00
14	{4, 2}, {5, 3}, {6, 2}, {15, 2}	22	1000	4.00
16	{5, 3}	27	750	3.00
15	{5, 4}, {8, 3}	15	700	5.00
17	{3, 1}, {6, 4}, {14, 1}	20	650	4.00
18	{9, 2}, {16, 3}	24	250	5.00
19	{4, 1}, {5, 5}, {6, 2}, {8, 2}, {15, 5}	5	450	1.00
20	{8, 2}, {11, 4}	11	50	5.00
21	{4, 5}, {5, 5}, {6, 2}, {8, 2}, {15, 5}	16	850	3.00
22	{12, 5}	24	200	5.00
23	{4, 2}, {5, 1}, {6, 5}, {8, 4}	14	50	4.00
24	{8, 4}, {11, 4}, {12, 5}, {13, 4}	7	200	5.00
25	{7, 3}, {10, 2}	24	350	1.00
26	{10, 2}	27	450	0.00
27	{8, 5}, {11, 5}, {12, 4}	22	400	1.00
28	{2, 1}, {8, 1}, {9, 2}	3	950	5.00
29	{4, 1}, {5, 5}	7	700	1.00
30	{11, 3}, {12, 5}	18	1000	1.00
31	{8, 2}, {10, 2}	2	800	2.00
32	{2, 3}, {6, 4}, {9, 3}	15	800	1.00
33	{5, 4}, {6, 5}, {15, 3}	27	500	4.00
34	{3, 2}, {6, 2}	12	300	4.00
35	{3, 4}, {14, 1}	9	900	2.00
36	{3, 2}	20	700	2.00
37	{1, 5}, {2, 2}, {6, 3}, {8, 3}, {9, 2}, {16, 4}	22	250	4.00
38	{2, 4}, {8, 3}, {9, 2}, {16, 5}	8	50	1.00
39	{6, 5}, {10, 5}	9	500	1.00
40	{2, 2}, {6, 4}, {9, 4}	7	250	5.00
41	{5, 1}, {8, 2}, {15, 1}	22	800	4.00
42	{2, 5}, {6, 4}, {9, 3}, {16,1}	19	400	2.00
43	{1, 3}, {5, 2}, {6, 2}, {8, 2}, {15, 3}	15	550	3.00
44	{2, 5}, {6, 4}, {9, 3}	12	350	1.00
45	{16, 3}, {8, 2}, {2, 3}, {9, 5}	15	400	3.00
46	{1, 3}, {12, 5}, {13, 4}	8	250	4.00

Table 1 Machining sequence, time, deadline, batch size, and penalty details (continuation)

Part No.	Processing sequence – {Machine No., Processing time (min)}	Deadline (days)	Batch size (Nos)	Penalty cost (INR/unit/day)
47	{13, 2}, {12, 3}	7	440	2.00
48	{8, 2}, {16, 3}, {5, 2}	10	350	2.00
49	{1, 3}, {11, 5}	9	300	1.00
50	{16, 2}, {9, 2}, {2, 1}, {6, 3}	8	300	1.00
51	{7, 3}, {10, 2}	20	250	2.00
52	{4, 1}, {1, 2}	16	300	3.00
53	{14, 3}	10	275	4.00
54	{10, 6}, {7, 2}	13	375	2.00
55	{16, 3}, {9, 4}, {6, 2}, {5, 3}	15	220	5.00
56	{13, 2}, {1, 7}, {11, 3}	12	200	3.00
57	{5, 3}, {6, 2}, {9, 3}, {2, 1}	5	150	1.00
58	{7, 5}	7	550	1.00
59	{10, 4}, {7, 8}	8	150	2.00
60	{2, 1}, {9, 3}, {16, 1}	17	500	1.00
61	{1, 6}, {13, 2}, {12, 3}	24	100	2.00
62	{11, 2}, {13, 4}	16	1000	2.00
63	{5, 3}, {2, 11}	18	240	3.00
64	{13, 2}, {11, 3}	27	800	1.00
65	{14, 3}, {3, 11}	19	440	2.00
66	{4, 4}, {1, 3}	14	320	2.00
67	{13, 2}, {1, 3}, {12, 4}, {11, 3}	22	600	4.00
68	{16, 2}, {9, 2}, {8, 1}, {6, 1}	14	700	1.00
69	{8, 1}, {9, 2}, {6, 3}, {5, 3}, {2, 2}	16	150	2.00
70	{7, 5}, {10, 1}	15	230	1.00
71	{3, 14}	7	450	2.00
72	{11, 6}, {12, 10}	18	570	3.00
73	{4, 1}, {1, 5}	9	250	4.00
74	{16, 3}, {9, 2}, {2, 2}	13	200	3.00
75	{16, 1}	3	230	1.00
76	{1, 2}, {5, 3}, {12, 1}	6	310	2.00
77	{2, 2}, {5, 1}, {6, 11}	12	330	3.00
78	{9, 3}, {6, 2}, {5, 3}	14	280	2.00
79	{2, 1}, {9, 3}	14	210	1.00
80	{8, 3}, {9, 3}	10	50	3.00

3.4 GA coding scheme and parameters, genetic operations

As the GA work on coding of parameters, the feasible job sequences (the parameters of the considered problems) are coded in two different ways and separately experimented for the same problem: fino-type coding and binary coding. In this work, fino-type coding is considered. In this coding each sequence is coded as 80 sets of two-digit numbers ranging from 01 to 80.

Example: 60, 54, 20, 79, 18, 45, 49, 72, 27, 41, 59, 34, 50, 32, 25, 29, 31, 2, 37, 69, 43, 21, 71, 67,46, 64, 6, 63, 19, 56, 74, 17, 15, 42, 35, 65, 1, 68, 52, 26, 7, 24, 57, 10, 75, 80, 28, 66, 36, 9, 13, 3, 4, 5, 30, 12, 16, 70, 55, 77, 76, 11, 14, 53, 48, 51, 58, 8, 22, 33, 73, 61, 62, 40, 44, 23, 78, 39, 47, 38.

GA parameters were:

- Population size: P = 100,
- Reproduction: tournament selection (target value – 0.75),
- Crossover probability: C = 0.6,
- Mutation probability: M = 0.01,
- Termination criteria: 3000 generations or a satisfactory value for objectives, whichever occurs first.

Consider the complexity of one iteration for the entire algorithm. The basic operations and their worst case complexities are as follows: $O(N^{(3/2)} \log N)$, where N is the number of bits in a single chromosome.

Reproduction

The tournament selection method is used for reproduction. Tournament selection is one of many methods of selection in genetic algorithms. Tournament selection involves running several "tournaments" among a few individuals chosen at random from the population. The winner of each tournament (the one with the best fitness) is selected for crossover. Selection pressure is easily adjusted by changing the tournament size. If the tournament size is larger, weak individuals have a smaller chance to be selected. Reproduction procedure is as follows:

- Selection method: tournament selection (assume the parameters for comparison as 0.75).
- Step 1: select two samples from the population.
- Step 2: evaluate the population.
- Step 3: generate random number in the range from 0 to 1.
- Step 4: if the random number is ≤ 0.75, select the best one, else select the inferior one.

Crossover

The strings in the mating pool formed after reproductions are used in the crossover operation (Fig. 5). Single-point crossover is used in this work. With a fino-type coding scheme, two strings are selected at random and crossed at a random site. Since the mating pool contains strings at random, we pick pairs of strings from the top of the list. When two strings are chosen for crossover, first a coin is flipped with a probability P_c = 0.6 check whether or not a crossover is desired. If the outcome of the coin flipping is true, the crossover is performed, otherwise the strings are directly placed in the intermediate population for subsequent genetic operation. Flipping a coin with a probability 0.6 is simulated using the Monte Carlo method. The next step is to find a cross site at random. Once crossover point is selected, till this point the permutation is copied from the first parent, then the second parent is scanned and if the number is not yet in the offspring it is added.

(**1 2 3 4 5** 6 7 8 9) + (**4 5 3 6 8** 9 7 2 1) = (**1 2 3 4 5** 6 8 9 7) (**4 5 3 6 8** 1 2 7 9)
Parent1 Parent2 Child1 Child2

Fig. 5 Crosssover operation

Mutation

The classic example of a mutation operator involves a probability that an arbitrary bit in a genetic sequence will be changed from its original state (Fig. 6). A common method of implementing the mutation operator involves generating a random variable for each bit in a sequence. This random variable tells whether or not a particular bit will be modified. The purpose of mutation in GAs is to allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution. This reasoning also explains the fact that most GA systems avoid only taking the fittest of the population in generating the next but rather a random (or semi-random) selection with a weighting toward those that are fitter. In this work, mutation probability is 0.01, i.e. 8 bits will be mutated. First generate random number from 0 to 1 with accuracy of 0.01. If random number is ≤ 0.01, then mutation is performed. The next step is to find a cross site at random, the two sites are selected by generating two random numbers between the numbers of jobs. For example, if the random numbers generated are 3 and 6, then the corresponding job numbers in these positions are exchanged.

1 2 **3** 4 5 **6** 7 8 9 ⟶ 1 2 **6** 4 5 **3** 7 8 9
Parent ⟶ Child

Fig. 6 Mutation operation

4. Results and discussions

The optimization procedures developed in this work are based on the modified non-dominated sorting genetic algorithm (NSGA-II). The FMS configuration considered in this work is taken from the literature [11]. In literature, procedure is developed for 43 jobs, using combined objective optimization method. A comparison between the proposed modified NSGA-II and other algorithms namely SPT, PSO, GA, CS [13] (found in literature) and NSGA-II has been presented in Table 2 and Fig. 7. But in this work we have taken the scheduling problem with 80 parts and multi objective optimization approach as well as modified NSGA-II is implemented. The result of modified NSGA-II and existing NSGA-II relating to the problem of 80 jobs are meticulously compared. Table 2 shows the results obtained by the proposed modified NSGA-II. It performs better in terms of objective functions and computational effort, i.e. 50 % less time than the NSGA-II. The Table 3 and Fig. 8 show the comparison of both the approaches in the study. The point in the graph shows the non-dominated points after 4500 generation using NSGA-II and modified NSGA-II.

Table 2 Comparison between various approaches

Algorithm	SPT [13]	PSO [13]	CS [13]	NSGA-II	Mod. NSGA-II
Machine idle time	180100	315650	163800	109850	95900
Penalty cost	101930	298196	138025	16298	10005
Sequence	20, 23, 38, 1, 9, 26, 22, 10, 34, 18, 36, 11, 25, 5, 16, 2, 40, 4, 41, 31, 7, 24, 28, 17, 6, 29, 35, 37, 15, 39, 42, 27, 33, 3, 43, 19, 13, 12, 32, 30, 8, 14, 21	27, 30, 38, 10, 18, 15, 34, 42, 5, 33, 8, 37, 23, 25, 9, 23, 5, 43, 20, 6, 4, 36, 19, 17, 24, 39, 31, 12, 8, 32, 26, 6, 14, 22, 3, 1, 11, 41, 9, 40, 21, 13, 7	8, 14, 28, 31, 3, 42, 26, 33, 22, 20, 5, 24, 2, 41, 18, 7, 10, 19, 23, 38, 4, 35, 40, 37, 15,17, 39, 6, 2, 34, 1, 29, 27, 16, 36, 30, 25, 32, 13, 3, 11, 10, 9	5, 30, 34, 28, 16, 24, 25, 10, 11, 27, 36, 2, 18, 1, 4, 29, 20, 13, 37, 17, 3, 9, 41, 12, 15, 6, 22, 7, 42, 38, 19, 23, 43, 21, 32, 14, 33, 8, 26,35,40,31,39	39, 34, 27, 11, 30, 22, 6, 16, 28, 23, 2, 26, 35, 7, 25, 43, 9, 40, 36, 41, 14, 37, 3, 42, 31, 18, 10, 24, 20, 17, 38, 21, 29, 4, 32, 15, 13, 33, 5, 1, 12, 8, 19

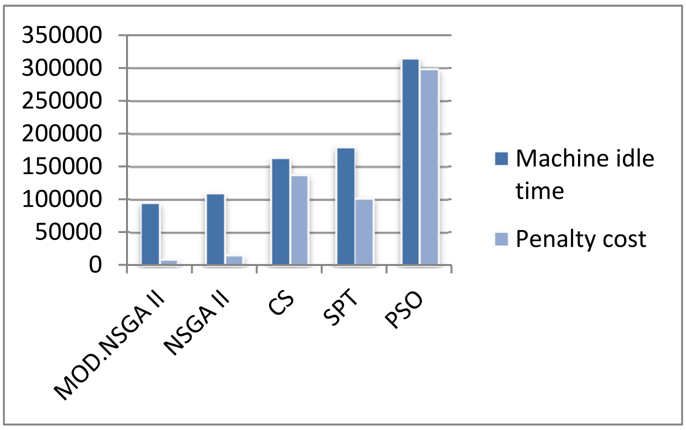

Fig. 7 Comparison between various approaches

Table 3 Results after 3000 generations (80 jobs scheduling problem)

Methodology	Trial No.	Machine idle time (min)	Minimum total penalty cost (INR)
NSGA-II	1	169835	98968.26
	2	167335	99770.35
	3	166415	100289.4
	4	166335	101909.6
	5	160485	108597.4
	6	159565	109116.5
Modified NSGA-II	1	121095	76757.08
	2	121095	76757.08
	3	129815	77308.82
	4	121065	77367.78
	5	121895	77645.97
	6	121895	77645.97
	7	126115	77719.44
	8	121095	77729.31
	9	119875	78162.64
	10	119675	78275.49
	11	127435	78321.32
	12	119545	78347.57
	13	119675	78358.47
	14	123145	78366.67
	15	120465	78409.79

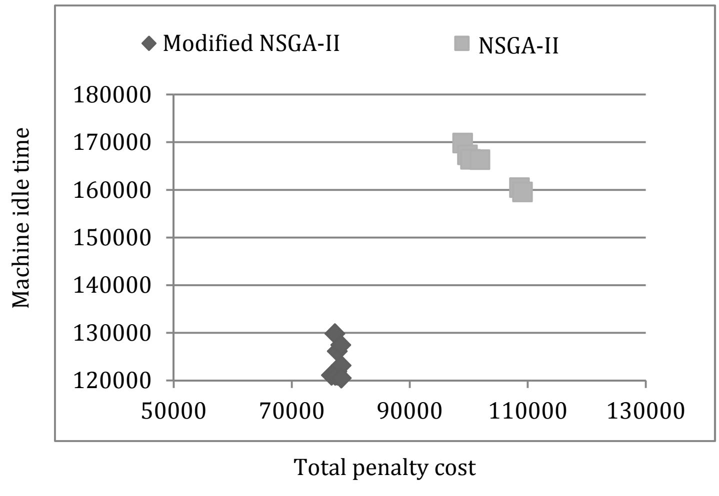

Fig. 8 Comparison of NSGA-II and modified NSGA-II

Results obtained for 80 jobs scheduling problem by modified NSGA-II

Global Pareto optimal front is obtained after executing 4500 generations and the details are shown in Table 4. Results are shown in Fig. 9. The software is executed on an Intel Core 2 Duo based PC with 4 GB RAM using .NET Framework. It took 15 min to complete the computation.

Table 4 Results after 4500 generations (80 jobs scheduling problem)

Methodology	Trial No.	Machine idle time (min)	Minimum total penalty cost (INR)
Modified NSGA-II	1	116625	73660.42
	2	111625	74692.36
	3	114175	74729.86
	4	114175	74901.74
	5	114175	74901.74
	6	114285	74916.32
	7	111625	75129.86
	8	111325	75207.99
	9	111505	75258.33
	10	114175	75790.63
	11	114175	75811.46
	12	111625	76003.13
	13	111505	76126.39
	14	113025	76270.49
	15	112675	76315.63
	16	114325	76545.83

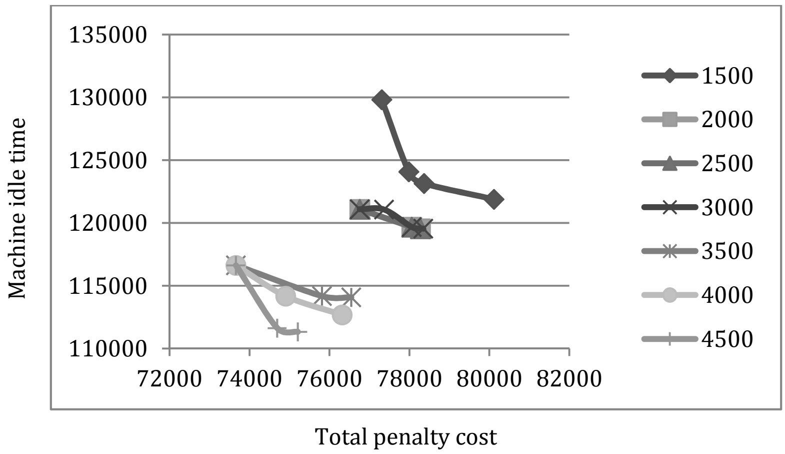

Fig. 9 Progression of Pareto-optimal fronts of modified NSGA-II

5. Conclusion

In this work the optimization procedure has been developed based on the modified multi-objective non-dominated genetic algorithm. This method is implemented successfully for solving the scheduling optimization problem of FMS. Software has been written in the .NET language. FMS schedule is obtained for 80 jobs and 16 machines. The result obtained by modified NSGA-II is analyzed for two objectives, i.e. minimizing total penalty cost and minimizing total machine idle time. After 4500 generation best solution is obtained. The computational effort of FMS scheduling problem is increasing proportional to the number of components. In case of 80 components 7.15694570462638022948115337231870e+118 combinations are possible. Due to very high computational effort exhaustive search is not possible. Similarly random search also requires so much of computational effort. By implementing genetic algorithm for 4500 generations $4.5 \cdot 10^5$ computations needed only for getting the optimal solution. In order to reduce the computational effort further, existing NSGA-II is modified. It is found that the proposed approach consumes 50 % time only in comparing with NSGA-II and is superior in terms of objective function. The procedure developed in this work can be suitably modified to any kind of FMS with a large number of components and machines. Future work will include the availability and handling time of loading and unloading stations, robots and AGV.

References

[1] Guo, Z.X., Wong, W.K., Leung, S.Y.S., Fan, J.T., Chan, S.F. (2008). A genetic-algorithm-based optimization model for scheduling flexible assembly lines, *The International Journal of Advanced Manufacturing Technology*, Vol. 36, No. 1-2, 156-168, doi: 10.1007/s00170-006-0818-6.

[2] Tiwari, M.K, Vidyarthi, N.K. (2000). Solving machine loading problems in flexible manufacturing system using a genetic algorithm based heuristic approach, *International Journal of Production Research*, Vol. 38, No. 14, 3357-3384, doi: 10.1080/002075400418298.

[3] Kumar, A.V.S.S., Veeranna, V., Prasad, B.D., Sarma, B.D. (2010). Optimization of FMS scheduling using non-traditional techniques, *International Journal of Engineering Science and Technology*, Vol. 2, No. 12, 7289-7296.

[4] Kumar, R., Tiwari, M.K., Shankar, R. (2003). Scheduling of flexible manufacturing systems: an ant colony optimization approach, In: *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, Vol. 217, No. 10, 1443-1453, doi: 10.1243/095440503322617216.

[5] Toker, A., Kondakci, S., Erkíp, N. (1994). Job shop scheduling under a non-renewable resource constraint, *The Journal of the Operational Research Society,* Vol. 45, No. 8, 942-947, doi: 10.2307/2584018.

[6] Hoitomt, D.J., Luh, P.B., Pattipati, K.R. (1993). A practical approach to job-shop scheduling problems, *Robotics and Automation: IEEE Transactions on*, Vol. 9, No. 1, 1-13, doi: 10.1109/70.210791.

[7] He, W., Kusiak, A. (1992). Scheduling manufacturing systems, *Computers in Industry,* Vol. 20, No. 2, 163-175, doi: 10.1016/0166-3615(92)90050-W.

[8] Lee, D.Y., DiCesare, F. (1994). Scheduling flexible manufacturing systems using Petri nets and heuristic search, *Robotics and Automation: IEEE Transactions on*, Vol. 10, No. 2, 123-132, doi: 10.1109/70.282537.

[9] Shnits, B., Sinreich, D. (2006). Controlling flexible manufacturing systems based on a dynamic selection of the appropriate operational criteria and scheduling policy, *International Journal of Flexible Manufacturing Systems,* Vol. 18, No. 1, 1-27, doi: 10.1007/s10696-006-9001-5.

[10] Yu, M.-C., Greene, T.J. (2006). A simulation analysis of controlling rules for flexible pull systems, *International Journal of Manufacturing Research*, Vol. 1, No. 3, 314-331, doi: 10.1504/IJMR.2006.011961.

[11] Jerald, J., Asokan, P., Prabaharan, G., Saravanan, R. (2004). Scheduling optimisation of flexible manufacturing systems using particle swarm optimisation algorithm, *The International Journal of Advanced Manufacturing Technology*, Vol. 25, No. 9-10, 964-971, doi: 10.1007/s00170-003-1933-2.

[12] Saravana Sankar, S., Ponnanbalam, S.G., Rajendran, C. (2003). A multiobjective genetic algorithm for scheduling a flexible manufacturing system, *The International Journal of Advanced Manufacturing Technology*, Vol. 22, No. 3-4, 229-236, doi: 10.1007/s00170-002-1464-2.

[13] Burnwal, S., Deb, S. (2013). Scheduling optimization of flexible manufacturing system using cuckoo search-based approach, *The International Journal of Advanced Manufacturing Technology*, Vol. 64, No. 5-8, 951-959, doi: 10.1007/s00170-012-4061-z.

[14] Udhayakumar, P., Kumanan, S. (2010). Sequencing and scheduling of job and tool in a flexible manufacturing system using ant colony optimization algorithm, *The International Journal of Advanced Manufacturing Technology,* Vol. 50, No. 9-12, 1075-1084, doi: 10.1007/s00170-010-2583-9.

[15] Kumar, M.V.S, Janardhana, R., Rao, C.S.P. (2011). Simultaneous scheduling of machines and vehicles in an FMS environment with alternative routing*, The International Journal of Advanced Manufacturing Technology,* Vol. 53, No. 1-4, 339-351, doi: 10.1007/s00170-010-2820-2.

[16] Saravanan, M., Haq, A.N. (2008). Evaluation of scatter-search approach for scheduling optimization of flexible manufacturing systems, *The International Journal of Advanced Manufacturing Technology*, Vol. 38, No. 9-10, 978-986, doi: 10.1007/s00170-007-1134-5.

[17] Nakamoto, F.Y., Miyagi, P.E., Santos Filho, D.J. (2008). Resources allocation control in flexible manufacturing systems using the deadlock avoidance method, In: *Proceedings of ABCM Symposium Series in Mechatronics,* Vol. 3, 454-460.

[18] Jian, C.F., Wang, Y. (2014). Batch task scheduling-oriented optimization modelling and simulation in cloud manufacturing, *International Journal of Simulation Modelling*, Vol. 13, No. 1, 93-101, doi: 10.2507/IJSIMM13(1)CO2.

[19] Reddy, B.S.P., Rao, C.S.P (2011). Flexible manufacturing systems modelling and performance evaluation using AutoMod, *International Journal of Simulation Modelling*, Vol. 10, No. 2, 78-90, doi: 10.2507/IJSIMM10(2)3.176.

[20] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, 182-197, doi: 10.1109/4235.996017.

[21] Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*, Wiley, New York, USA.

[22] Nidhiry, N.M., Saravanan, R. (2014). Scheduling optimization of FMS using NSGA-II, *Advances in Industrial Engineering and Management*, Vol. 3, No. 1, 63-72, doi: 10.7508/AIEM-V3-N1-63-72.

[23] Nidhiry, N.M., Saravanan, R. (2014). FMS scheduling optimization using modified NSGA-II, *International Journal of Mechanical and Production Engineering*, Vol. 2, No. 2, 1-6.