Measurement 114 (2018) 78–90

Contents lists available at ScienceDirect

Measurement

journal homepage: www.elsevier.com/locate/measurement

An effective hybrid multi objective evolutionary algorithm for solving real time event in flexible job shop scheduling problem

M.B.S. Sreekara Reddy[a], Ch. Ratnam[b], G. Rajyalakshmi[c], V.K. Manupati[d,*]

[a] Department of Mechanical Engineering, K L University, Green Fields, Vaddeswaram, Guntur district, AP, India
[b] Dept. of Mechanical engineering, AUCE, Visakha Patnam, AP, India
[c] Department of Manufacturing Engineering, VIT University, Vellore, Tamil Nadu, India
[d] Department of Mechanical Engineering, VIT University, Vellore, Tamil Nadu, India

ARTICLE INFO

Keywords:
Flexible job shop
Multi-objective evolutionary algorithm
NP-hard
Optimization

ABSTRACT

This paper addresses the multi-objective model for a flexible job shop scheduling problem (FJSSP) to improve the system performance under the condition of machines break down as a real time event. It is important to identify the relevant performance measures to the mentioned problem for examining the system performance. Therefore, minimization of make span and minimization of total machine load variation is considered as two performance measures. Generally, it is very difficult to develop a mathematical model for the real-time situations in FJSSP. Hence, in this paper we divided the research work into two folds: Primarily, a mixed-integer non-linear programming (MINLP) model has been developed to represent the above-mentioned multi-objectives that subjected to constraints without considering machines break down. Secondarily, by incorporating the machines break down as the real-time event the performance of the system is examined. Solving conflicting objectives simultaneously for finding the optimal/near optimal solutions in a reasonable time is a challenge. In this paper, we proposed a new evolutionary based multi-objective teacher learning-based optimization algorithm (MOTLBO) to solve the above-mentioned complex problem. Moreover, to improve the obtained solutions a local search technique has been incorporated in the MOTLBO and comparisons has been made with existing multi-objective particle swarm optimization (MOPSO) and conventional non-dominated sorting genetic algorithm (CNSGA-II). Results found that the proposed multi-objective-based hybrid meta-heuristic algorithm produced high-quality solutions as proved by the tests we performed over a number of randomly generated test problems. Finally, comparisons also made with how the machines break down can affect the proposed systems performance.

1. Introduction

Current manufacturing environment has to change its trend due to recent advancements in information and communication technology, the requirement of customized products from customers, and huge competition between the industries. To cater these mentioned requirements, particularly in shop floor, it is necessary to choose effective manufacturing strategies that need to integrate manufacturing functions such as process planning and scheduling effectively and efficiently. In recent years, many researchers try to achieve the feasible scheduling with extension version of classical job shop scheduling problem (JSSP) i.e. FJSSP due to its wide variety of applications.

In JSSP, processing of each operation for a job can be only possible on one machine, but in the case of FJSSP, each operation of a job can be processed on several machines for a given set [1]. Therefore in FJSSP, it is very hard to identify the assigned operations on a particular machine out of a set of competent machines and then identify the sequence of operations is a complex task and it is much complex than JSSP. Nouri et al. [30] generalized the classical job shop scheduling problem by integrating the transportation times and many robots problem where a set of jobs are processed by a set of alternative machines and transported by several robots. From the literature, job shop scheduling problem is known as a NP-hard problem [2] thus, it can be concluded that FJSSP as NP-hard combinatorial optimization problems.

In general, FJSSPs have been solving by two different approaches i.e. hierarchical and integrated approach [3,10]. In hierarchical/traditional approach, process planning and scheduling have been conducted in a sequential manner. In other words, to solve the problem with the

* Corresponding author.
E-mail addresses: mbssreddy@kluniversity.in (M.B.S. Sreekara Reddy), chratnam@gmail.com (C. Ratnam), rajyalakshmi@vit.ac.in (G. Rajyalakshmi), manupativijay@gmail.com (V.K. Manupati).

http://dx.doi.org/10.1016/j.measurement.2017.09.022
Received 14 May 2016; Received in revised form 27 April 2017; Accepted 12 September 2017
Available online 14 September 2017
0263-2241/

hierarchical approach, divide the problem into sub-problems and then solve it individually [4]. Javid and Hooshangi [29] addressed a ternary-integration scheduling problem which incorporated employee time-tabling into the schedules of machines and transportation systems in a job shop environment and also proved that the effectiveness and efficiency of Anarchic Society Optimization (ASO) in the integrated job-shop scheduling problem. The greatest difficulty with this approach is there might be a chance of losing good quality solutions and obtaining a compromising solution for both the objectives. (Saygin and Kilc, 1999) mentioned various difficulties with the traditional/hierarchical approach and the obstacles to improve the productivity of the manufacturing systems. Recently, researchers have been developed many methods and approaches for integrating the manufacturing functions to improve the performance of the system [32]. Khoukhi et al. [27] focused on the FJSSP with machine unavailability constraints for proceeding preventive maintenance activities. With the objective of minimizing the make-span, they proposed the formulations of a mixed integer nonlinear program and a bi-level disjunctive/conjunctive graph.

Different approaches to solving the FJSSP were proposed by [5,6]. Yang [7] proposed the multistage genetic algorithm to solve the FJSSP for improving the performance measures of the system. The complex problem of FJSSP with the consideration of controllable processing times CPT) was explored by [28]with the main aim of minimizing the makespan and the total additional resource consumption. Tay and Ho [8] combined different dispatching rules as composite rules for solving the multi-objective FJSSP by improving the objective functions such as minimization of makespan, mean tardiness, and mean flow time. With sensitivity analysis, the validation of the presented rules has been conducted and found the proposed approach robustness. Wang et al. [9]) presented the bi-population based estimation of distribution algorithm to solve the FJSSP. Here, with Taguchi method, the parameters have been investigated and then identified the better ones. With these identified parameters the algorithm performance has been examined and found the best performance. Kaplanoğlu [31] provided an object oriented (OO) approach for solving the multi objective FJSSP by using a simulated annealing optimization algorithm. Over the past few years, much research and study [16,12,13,14] are done by many researchers and they focused on different hybrid evolutionary algorithms to solve the multi-objective FJSSP. Moreover, to our knowledge, a real-time event such as machine breakdown in FJSSP is difficult to be expressed and it has not been studied well.

In this paper, an effective hybrid meta-heuristic (HMOTLBO) algorithm is introduced to solve the real time event in multi-objective FJSSP. To carry out the multi-objective FJSSP, we have carried out minimization of make span, and minimization of total machines workload variation as performance measures and then developed a mixed integer non-linear programming (MINLP) based mathematical model to represent the FJSSP. Moreover, we have introduced the machine breakdowns randomly in the system as a real-time event to examine the proposed manufacturing system. Finally, with different instances the proposed HMOTLBO algorithm is implemented and solved, comparisons is made with well-established multi-objective evolutionary algorithms, i.e., CNSGA-II, MOPSO. Finally, with results we proved that the proposed algorithm outperforms other compared algorithms and also examine the system behaviour at real time situations.

The remainder of this research work is organised as follows: Section 2 introduced the problem description with basic assumptions and developed a multi-objective-based mathematical model along with the constraints. In Section 3, CNSGA-II, MOPSO algorithms are discussed and proposed hybrid MOTLBO is detailed for solving the proposed FJSSP problem. Experiments with different problem instances are illustrated in Section 4. Section 5, discussed obtained results of the proposed method. Finally, conclusions are drawn and future work is suggested in Section 6.

2. Problem description

FJSSP is addressed in this research study that consists of a set of jobs j with k operations where k varies between $(1 \geqslant k \geqslant n)$. Each operation k_{jm} of jobs j is to be processed on one machine m from the set of eligible machines M_{kj}. It is assumed that the processing times of the operations to process on machines is known well in advance and all machines are continuously available at time zero i.e. before scheduling of operations. Each machine can only process one operation at a time. Thus the consecutive jobs can wait at buffers until its preceding job can finish its process. Therefore, the buffers are considered as unlimited in size. The setup time is considered and it is included in the processing time, and the transportation time between two corresponding machines is assumed to be zero. In this paper, we considered two performance measures such as minimization of makespan and workload of machines with the intention of protecting machines from overuse. A mathematical model has been developed with considered multi-objectives by assuming the real-time event as machine breakdowns and repairs. In this study, it is assumed that the real time event occurs randomly which follows an exponential distribution and the machines selected at the beginning of the simulation can be termed as key machines. If the key machine fails its process, then it is assumed that the real-time event occurs. Here, we consider the mean time between failures (MTBF) and mean time to repair (MTTR) are two system parameters useful for the analysis of the system effectiveness when machine breakdown exist [33]. However, to meet the requirements from the real world production based multi-objective problem is considered to be a new problem according to current manufacturing scenario. From the literature it is clear that the FJSSP is NP-hard in nature, to find compromising solution for the mentioned problem in a reasonable time is a challenge. Thus an effective evolutionary algorithm based approach is proposed for enhancing the performance of the system.

Assumptions:

(1) The considered machines before scheduling must be available at time zero.
(2) All jobs can be started at time zero.
(3) At a time each machine can process only one operation.
(4) Processing of operations on the machines should not be interrupted except when machine breakdown.
(5) The sequence of operations of each job to further process is pre-specified.
(6) Neither release times nor due dates are specified.
(7) Job transportation time among machines is not considered.

Before describing the mathematical model, it is better to define some symbols and notations are as follows:

Z The number of operations for each job as a set.
If $z(j) = k$, it defines that job j has k operations.
$C_{z(j)jk}$ Completion time of Kth operation of job j on machine k
T_{rjk} Processing time of rth operation of job j on machine k
T_k The sum of the time that machine k is doing process

Objective Functions:

$$F1\colon\ Min\max\{C_{z(j)jk}\};\ z(j) \in Z;\ j \in N;\ k \in M \tag{1}$$

$$F2\colon\ Min\ \max\left\{T_r = \sum_{j=1}^{n}\ \sum_{l=1}^{z(j)}\ t_{ljk}\right\} \tag{2}$$

Subject to:

$$c_{ljk} - t_{ljk} \geqslant c_{(l-1)jk}\ l = 1,2,3,\ldots,z(j);\ j = 1,2,3,\ldots,n;\ k = 1,2,\ldots,m \tag{3}$$

$$c_{ljk} - t_{ljk} \geqslant c_{lij};\ l = 1,2,\ldots,z(j);\ i = 1,2,\ldots,n;\ j = 1,2,\ldots,m \tag{4}$$

$$c_{ljk} \geqslant 0; l = 1,2,...,z(j); j = 1,2,...,n; k = 1,2,...,m \tag{5}$$

The above-mentioned objectives, i.e. minimization of make-span ($F1$) and minimization of workload on the most loaded machine ($F2$) are given by Eqs (1) and (2). Constraint (3) and (4) indicates processing constraints where the preceding constraints among operations of the same job should follow and each machine should be available to other operations if the concerned operations complete.

3. Multi-objective evolutionary algorithm

A comprehensive study on multi-objective optimization by meta-heuristics was first discussed (Andrej 2001). Schaffer (1985) proposed the vector evaluated genetic algorithm (VEGA) which is a modified version of the single-objective genetic algorithm. But, Coello (1999) observed that the non-dominated solutions produced from VEGA are almost concentrating on the middle region of the Pareto front, rather than spread in the Pareto fronts. Fleming et al. [34], proposed a method to overcome the mentioned difficulties, one can find the detailed information of the recent multi-objective optimization for conflicting objectives and how these algorithms can work effectively and efficiently can be found in paper of [35]. In our research work, we have used conventional multi-objective-based non-dominated sorting genetic algorithm as a benchmark and recently developed MOPSO algorithm. Later, with proposed hybrid MOTLBO both the algorithms is compared. The detail description of the mentioned algorithms is as follows:

3.1. Non-dominated sorting genetic algorithm (NSGA-II)

The Non-dominated sorting genetic algorithm has been considered as one of the best algorithms to solve multi-objective problems. It has operators such as non-dominated sorting and crowding distance to improve the solution qualities by considering both the objectives simultaneously. Goldberg et al. [25] and Srinivas and Deb, (1995) highlighted various drawbacks of the non-dominated sorting genetic algorithm (NSGA) and suggested the improved version of NSGA such as NSGA-II. In this work, we adopted NSGA-II to solve the proposed problem and the details of the Schematic procedure for processing NSGA-II is shown in Fig. 1.

3.2. Multi-objective particle swarm optimization algorithm (MOPSO)

In the area of manufacturing and production, particularly for FJSS based single objective optimization problems, population-based particle swarm optimization PSO) algorithm has been used for obtaining an optimal solution. Its evolution lies from the concepts of social behaviour of bird flocking and fish schooling. However, many researchers implemented this algorithm in a wide range of applications. Kennedy and Eberhart [15]) briefed a variety of applications that has been solved with particle swarm optimization algorithm. Xia and Wu [11] adopted a linear weighted summation approach in which aggregation of three objective values has been simultaneously solved with hybrid PSO and simulated annealing (SA) algorithm. Here, PSO strives for searching of good routing decisions, and SA algorithm explore good sequencing decisions which leads to finding better optimal solutions. However, to solve for multi-objective problems and to obtain best compromising solutions the existing PSO algorithm has to be modified. Recently, a Pareto archive PSO was developed for the multi-objective FJSSP to obtain a set of Pareto optimal solutions [18] Zhang et al. [17] developed an improved GA combined with tabu search (TS) and also PSO combined with TS to solve the multi-objective FJSSP.

In this research work, we have incorporated non-dominated sorting and crowding distance operators that may lead to high speed of convergence to find the multi-objective solutions. The schematic procedure and the flowchart of MOPSO is shown as follows:

Step 1 Initialize swarms as random population.
Step 2 Calculate the velocities for every particle.
Step 3 Set of leaders is initialized with solutions obtained from non-dominated sorting operators and those are stored in the external archive.
Step 4 Quality measures such as velocity, position and cognitive learning factor are considered for selecting the leaders in the archive.

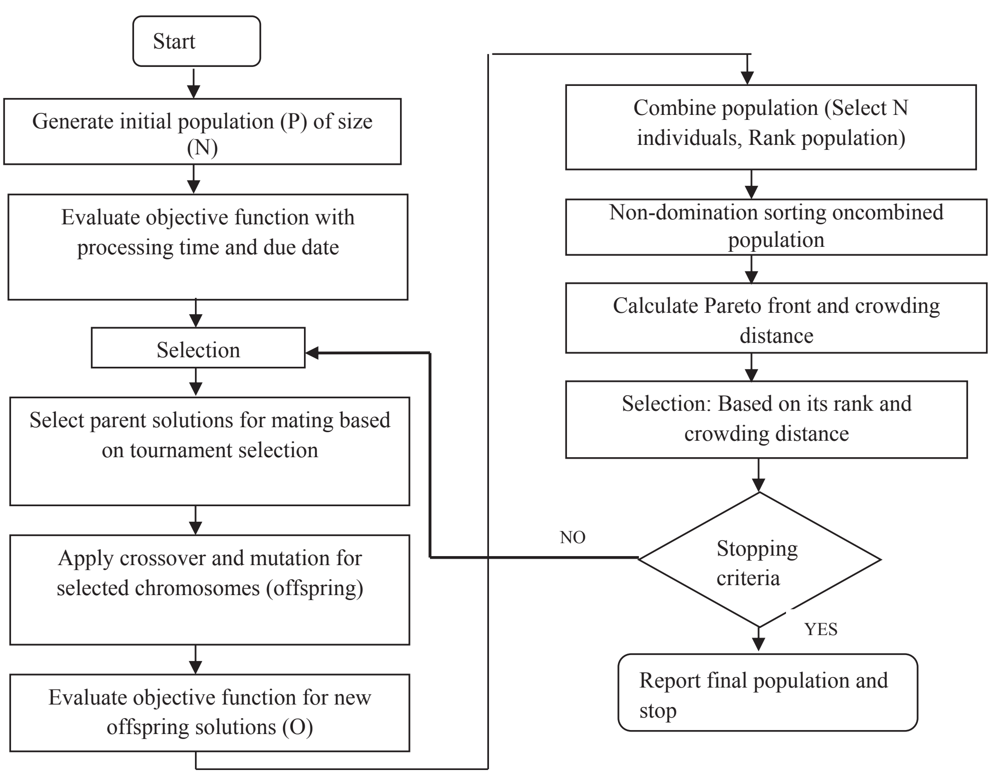

Fig. 1. Flow Chart of conventional NSGA-II.

Table 1
Input data of the FJSSP of 10 by 10 instance [4].

Jobs/Machines		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10
J1	Oper1	1	4	6	9	3	5	2	8	9	5
	Oper 2	4	1	1	3	4	8	10	4	11	4
	Oper 3	3	2	5	1	5	6	9	5	10	3
J2	Oper 1	2	10	4	5	9	8	4	15	8	4
	Oper 2	4	8	7	1	9	6	1	10	7	1
	Oper 3	6	11	2	7	5	3	5	14	9	2
J3	Oper 1	8	8	8	9	4	3	5	3	8	1
	Oper 2	9	3	6	1	2	6	4	1	7	2
	Oper 3	7	1	8	5	4	9	1	2	3	4
J4	Oper 1	5	10	6	4	9	5	1	7	1	6
	Oper 2	4	2	3	8	7	4	6	9	8	4
	Oper 3	7	3	12	1	6	5	8	3	5	2
J5	Oper 1	7	10	4	5	6	3	5	15	2	6
	Oper 2	5	6	3	9	8	2	8	6	1	7
	Oper 3	6	1	4	1	10	4	3	11	13	9
J6	Oper 1	8	9	10	8	4	2	7	8	3	10
	Oper 2	7	3	12	5	4	3	6	9	2	15
	Oper 3	4	7	3	6	3	4	1	5	1	11
J7	Oper 1	1	7	8	3	4	9	4	13	10	7
	Oper 2	3	8	1	2	3	6	11	2	13	3
	Oper 3	5	4	2	1	2	1	8	14	5	7
J8	Oper 1	5	7	11	3	2	9	8	5	12	8
	Oper 2	8	3	10	7	5	13	4	6	8	4
	Oper 3	6	2	13	5	4	3	5	7	9	5
J9	Oper 1	3	9	1	3	8	1	6	7	5	4
	Oper 2	4	6	2	5	7	3	1	9	6	7
	Oper 3	8	5	4	8	6	1	2	3	10	12
J10	Oper 1	4	3	1	6	7	1	2	6	20	6
	Oper 2	3	1	8	1	9	4	1	4	17	15
	Oper 3	9	2	4	2	3	5	2	4	10	23

Table 2
Initialization of the data variables.

	M/c 1	M/c 2	M/c 3	M/c 4	M/c 5	M/c 6	M/c 7	M/c 8	M/c 9	M/c 10	f(x1)
J1	2.71	1.92	4.75	8.91	4.14	5.93	3.57	4.59	9.57	3.38	49.47
J2	2.67	8.10	2.58	3.91	4.52	6.07	7.47	10.51	10.35	3.80	70.65
J3	3.30	9.59	5.63	4.39	6.69	7.77	6.66	9.38	8.19	2.06	114.77
J4	4.93	8.80	4.54	3.82	5.45	4.77	4.62	12.68	7.64	1.77	174.87
J5	5.95	9.73	2.21	2.15	4.58	5.47	2.81	13.55	8.90	1.88	234.01
J6	7.88	10.27	7.74	8.19	4.57	5.90	4.63	7.54	7.85	1.45	290.82
J7	8.71	2.49	7.42	2.23	2.07	7.16	3.39	2.21	4.50	3.43	358.81
J8	8.12	7.31	6.71	1.78	6.13	5.70	2.87	5.84	6.48	5.33	404.33
J9	5.77	3.66	6.81	7.29	6.18	6.97	3.16	2.55	3.62	5.22	460.46
J10	4.56	6.03	4.34	1.47	7.34	4.72	5.25	5.95	3.33	5.20	511.69
Mean	5.46	6.79	5.27	4.41	5.17	6.05	4.44	7.48	7.04	3.35	
f(x2)	4.50	8.72	3.35	6.88	2.08	0.89	2.34	14.22	5.54	2.10	

Table 3
Population group in a teaching phase for minimization of makespan.

Teacher 1	2.71	1.92	4.75	8.91	4.14	5.93	3.57	4.59	9.57	3.38	49.47
Teacher 2	2.67	8.10	2.58	3.91	4.52	6.07	7.47	10.51	10.35	3.80	70.65
Teacher 3	3.30	9.59	5.63	4.39	6.69	7.77	6.66	9.38	8.19	2.06	114.77
Teacher 4	5.95	9.73	2.21	2.15	4.58	5.47	2.81	13.55	8.90	1.88	234.01
Teacher 5	8.12	7.31	6.71	1.78	6.13	5.70	2.87	5.84	6.48	5.33	404.33

Step 5 For each iteration update velocity and position of each particle in the swarm.
Step 6 Evaluate fitness function to sort *pbest* and *gbest*.
Step 7 Implemented non-dominated sorting and crowding distance operators to find the set of optimal solutions spread across the optimal Pareto front.
Step 8 Update the solutions in the external archive to obtain optimal *gbest*.
Step 9 Check the termination criteria. Otherwise, return to step 2.

3.3. Hybrid multi-objective teacher learning based optimization (MOTLBO)

In this paper, we developed a new kind of optimization approach to solve the proposed multi-objective optimization problem called multi-

Table 4
Population group in teaching phase for Minimization of m/c workload.

Teacher 1	Teacher 2	Teacher 3	Teacher 4	Teacher 5
5.93	4.14	4.75	9.57	1.92
6.07	4.52	2.58	10.35	8.10
7.77	6.69	5.63	8.19	9.59
4.77	5.45	4.54	7.64	8.80
5.47	4.58	2.21	8.90	9.73
5.90	4.57	7.74	7.85	10.27
7.16	2.07	7.42	4.50	2.49
5.70	6.13	6.71	6.48	7.31
6.97	6.18	6.81	3.62	3.66
4.72	7.34	4.34	3.33	6.03
0.89	2.08	3.35	5.54	8.72

objective-based teacher learning based algorithm to obtain Pareto optimal solutions. Recent works on TLBO [19–22]; [23,24], proved that it has the capacity to solve wide variety of problems effectively and efficiently for enhancing the systems performance. But, till date very limited applications with multi-objective-based teacher learning optimization has been tested. In this research work, we adopted the Pareto approach to the TLBO algorithm and then proposed a multi-objective population based TLBO algorithm to solve the MOFJSSP. Moreover, we adopted a local search technique to improve the convergence rate and quality of the solutions. The detailed step-wise procedure of the proposed algorithm and its approach is described as follows:

Step 1 **Initialization** *Stage:* Initialize the population (learners), parameters of population size, design variables (number of subjects offered to the learners) randomly with threshold values, and termination criterion.

$$\alpha^{1}_{(i,j)} = \alpha_j^{\min} + \text{rand}_{(i,j)} \times (\alpha_j^{\max} - \alpha_j^{\min}) \tag{6}$$

The initialization starts with randomly generated values, and it is represented with *N* rows and *K* columns as the population size of the class and a total number of offered subjects. The Eq. (6) represents the assigned randomly generated values for initial generation of *i*th vector to the *j*th parameter. Where α represents the particular student *i* and rand *(i, j)* represents uniformly distributed random variable ranging from 0 to 1. Eq. (7) indicates the parameters of the *i*th vector for the generation *g*.

$$Z^{g}_{(i)} = [\alpha^{g}_{(i,1)}, \alpha^{g}_{(i,2)}, \ldots, \alpha^{g}_{(i,j)}, \ldots, \alpha^{g}_{(i,D)}] \tag{7}$$

$$\begin{bmatrix} \beta a_i^{g} \\ \beta b_i^{g} \end{bmatrix} = \begin{bmatrix} fa(Z^{g}_{(i)}) \\ fb(Z^{g}_{(i)}) \end{bmatrix} \tag{8}$$

In this paper, we have carried out a multi-objective problem with *a* and *b* as two objectives represented in Eq. (8). The Table 1 represents an FJSSM problem of a manufacturing industry which has ten jobs undergoing ten machining operations where each job can be completed in any of the three sequences of operation. Table 2 is the initialized table which is derived by substituting the values of Table 1 in Eq. (6).

Step 2: Evaluate the knowledge of the learners as fitness values.

In any evolutionary algorithm premature convergence is avoided by one of a powerful mechanism called as population sorting. In MOTLBO algorithm, we divide the population into a number of groups and each groups having individual teachers. Thus population sorting mechanism is provided by splitting the population into respective groups and adapting multi-teacher concept. The multi-teacher concept selects the best learner of each group as a teacher of its respective group and this teacher will be responsible for increasing the mean of the group and trying to bring mean to its level. Once the group mean reaches the level of the learner the entire group reaches the next group. The learner with least f(X) value is selected as chief teacher and remaining teachers of the group is selected by the Eq. (9).

$$T_s = \text{f}(\text{X}^{b}) \neq \text{r}_i * \text{f}(\text{X}^{b}) \quad \text{s}= 2,3,\ldots\ldots\ldots,\text{N} \tag{9}$$

Step 3: Adapting teacher factor and assigning learners to the teacher.

In TLBO, teacher factor used to be either 1 or 2 that is either students learns everything from the teacher or nothing from the teacher, In real life situation the teacher-learner phenomenon usually varies between 1 to 2 not either 1 or 2.So the earlier equation has to be modified in such a way that we get near optimum value for teacher factor and this modified equation is been represented in Eq. (10).

$$\begin{aligned} (T_F)_{s,i} &= \left(\frac{f(X^{k})}{T_s}\right) \quad \text{if } T_s \neq 0 \\ (T_F)_i &= 1 \qquad\qquad \text{if } T_s = 0 \end{aligned} \tag{10}$$

where $\text{f}(\text{X}^{\text{k}})$ is a result of any learner *k* associated with any group *'s'* considering of *i* iteration of all subjects and T_s is the teacher of the same group in the same iteration.

Assigning the learners is done by simple calculation that is shown below.

For k =1 to Population
If $\text{T}_1 \le \text{f}(\text{X}_\text{k}) < \text{T}_2$
Assign the learner $\text{f}(\text{X}_\text{k})$ to teacher 1 (i.e., T_1).
Else If $\text{T}_2 \le \text{f}(\text{X}_\text{k}) < \text{T}_3$
Assign the learner $\text{f}(\text{X}_\text{k})$ to teacher 2 (i.e., T2)
.
.
.

Table 5
Assignment of groups for Minimization of makespan.

Group 1	J1	2.71	1.92	4.75	8.91	4.14	5.93	3.57	4.59	9.57	3.38
	Mean	2.71	1.92	4.75	8.91	4.14	5.93	3.57	4.59	9.57	3.38
Group 2	J2	2.67	8.10	2.58	3.91	4.52	6.07	7.47	10.51	10.35	3.80
	Mean	2.67	8.10	2.58	3.91	4.52	6.07	7.47	10.51	10.35	3.80
Group 3	J3	3.30	9.59	5.63	4.39	6.69	7.77	6.66	9.38	8.19	2.06
	J4	4.93	8.80	4.54	3.82	5.45	4.77	4.62	12.68	7.64	1.77
	Mean	4.11	9.20	5.08	4.10	6.07	6.27	5.64	11.03	7.91	1.92
Group 4	J5	5.95	9.73	2.21	2.15	4.58	5.47	2.81	13.55	8.90	1.88
	J6	7.88	10.27	7.74	8.19	4.57	5.90	4.63	7.54	7.85	1.45
	J7	8.71	2.49	7.42	2.23	2.07	7.16	3.39	2.21	4.50	3.43
	Mean	7.51	7.49	5.79	4.19	3.74	6.18	3.61	7.77	7.08	2.25
Group 5	J8	8.12	7.31	6.71	1.78	6.13	5.70	2.87	5.84	6.48	5.33
	J9	5.77	3.66	6.81	7.29	6.18	6.97	3.16	2.55	3.62	5.22
	J10	4.56	6.03	4.34	1.47	7.34	4.72	5.25	5.95	3.33	5.20
	Mean	6.15	5.67	5.95	3.51	6.55	5.79	3.76	4.78	4.48	5.25

Table 6
Assignment of groups for Minimization of m/c workload.

Group 1		Group 2				Group 3			Group 4			Group 5		
	Mean				Mean			Mean			Mean			Mean
5.93	5.93	4.14	3.38	3.57	3.70	4.75	2.71	3.73	9.57	8.91	9.24	1.92	4.59	3.26
6.07	6.07	4.52	3.80	7.47	5.26	2.58	2.67	2.62	10.35	3.91	7.13	8.10	10.51	9.31
7.77	7.77	6.69	2.06	6.66	5.14	5.63	3.30	4.46	8.19	4.39	6.29	9.59	9.38	9.48
4.77	4.77	5.45	1.77	4.62	3.95	4.54	4.93	4.73	7.64	3.82	5.73	8.80	12.68	10.74
5.47	5.47	4.58	1.88	2.81	3.09	2.21	5.95	4.08	8.90	2.15	5.53	9.73	13.55	11.64
5.90	5.90	4.57	1.45	4.63	3.55	7.74	7.88	7.81	7.85	8.19	8.02	10.27	7.54	8.90
7.16	7.16	2.07	3.43	3.39	2.96	7.42	8.71	8.06	4.50	2.23	3.36	2.49	2.21	2.35
5.70	5.70	6.13	5.33	2.87	4.78	6.71	8.12	7.41	6.48	1.78	4.13	7.31	5.84	6.57
6.97	6.97	6.18	5.22	3.16	4.85	6.81	5.77	6.29	3.62	7.29	5.46	3.66	2.55	3.11
4.72	4.72	7.34	5.20	5.25	5.93	4.34	4.56	4.45	3.33	1.47	2.40	6.03	5.95	5.99

Table 7
Distance Matrix in a teaching phase for minimization of makespan.

Group	M.D									
1	2.71	1.92	4.75	8.91	4.14	5.93	3.57	4.59	9.57	3.38
2	2.67	8.10	2.58	3.91	4.52	6.07	7.47	10.51	10.35	3.80
3	−1.76	−3.66	−1.08	−1.06	−1.14	−0.79	−0.80	−6.39	−1.91	−0.29
4	−0.47	0.25	−1.02	−1.87	−0.05	−0.83	−1.13	2.80	0.05	−0.39
5	0.65	0.76	−0.05	−0.73	−0.65	−0.40	−0.53	0.08	0.87	−0.34

Table 8
Distance Matrix in a teaching phase for minimization of m/c workload.

M.D				
1	2	3	4	5
5.93	−0.01	−0.17	−1.79	−3.01
6.07	−1.15	−0.66	0.31	−6.39
7.77	0.37	−0.27	0.21	−4.59
4.77	0.87	−0.56	0.51	−1.81
5.47	0.19	−0.14	1.65	−4.75
5.90	0.02	−0.16	−1.85	−3.40
7.16	−1.21	−0.68	0.14	−0.67
5.70	0.25	−2.45	1.18	−0.02
6.97	0.35	−0.61	−3.02	−0.39
4.72	0.27	−0.89	0.05	−0.69

Else If $T_{N-1} \leq f(Xk) < T_N$
  Assign the learner $f(X_k)$ to teacher $N-1$(i.e. T_{N-1})
Else
  Assign the learner $f(X_k)$ to teacher T_N .
End if
End For

The above algorithm illustrates how different students based on their performance can be allotted to different teachers if the learning capacity for every student k i.e. $f(X^k)$ is less than that of the particular teacher who teaches the subject. This is followed by allotting that student to that particular teacher for the purpose of his understanding and development. This process continues till each and every student is allotted based on their performance in various subjects under various teachers as per the requirement. Finally, the last student who is left is well versed in all the subjects taught by teachers from 1 to $N-1$ and thus by default is assigned the *Nth* teacher. For example if the student 1 is weak in a subject taught by the teacher 1 as compared to the subject taught by teacher 2, the algorithm will allot the student under teacher 1.

The tables shown below represent teachers of the respective group and the grouping of the population. The information of each table is mentioned below. Population group in a teaching phase for minimization of make-span is presented in Table 3 and is derived from the entries of Table 2 which contains make-span values for each teacher and their calculated fitness value based on fitness function f(x1). Table 4 presents the machine load population group in a teaching phase derived from the entries in Table 2 containing the machine workload values for each teacher and their calculated fitness value based on fitness function f (x2). Table 5 contains the groups that are made within the jobs and their corresponding make-span values for each job and their mean for each group. Similarly, Table 6 contains the groups that are made within the entries of the machines for their respective workload values and

Table 9
X_{new} values generated after learner phase for make span.

X-New final											
	M/c 1	M/c 2	M/c 3	M/c 4	M/c 5	M/c 6	M/c 7	M/c 8	M/c 9	M/c 10	f(x)
J1	2.71	1.92	4.75	8.91	4.14	5.93	3.57	4.59	9.57	3.38	49.47
J2	2.67	8.10	2.58	3.91	4.52	6.07	7.47	10.51	10.35	3.80	70.65
J3	1.53	5.93	4.55	3.32	5.54	6.98	5.87	2.99	6.28	1.77	111.21
J4	1.63	5.70	3.63	3.32	5.47	6.91	5.75	5.03	5.85	1.63	155.85
J5	5.49	9.98	1.19	0.28	4.53	4.64	1.68	16.35	8.95	1.49	200.63
J6	5.63	10.50	6.23	2.55	4.52	4.66	2.22	11.60	8.87	1.43	255.15
J7	7.19	4.85	6.24	0.30	2.37	4.68	1.78	14.11	5.75	2.29	314.21
J8	8.77	8.07	6.66	1.05	5.47	5.29	2.34	5.91	7.34	4.99	366.48
J9	8.41	5.37	6.68	3.38	5.51	5.91	2.43	4.89	5.87	4.94	422.32
J10	7.42	7.76	6.45	0.82	5.61	5.15	4.58	5.96	5.95	4.87	475.65

Table 10
X_{new} values generated after learner phase for machine load.

X-New final										
	M/c 1	M/c 2	M/c 3	M/c 4	M/c 5	M/c 6	M/c 7	M/c 8	M/c 9	M/c 10
J1	2.72	1.92	4.57	7.43	4.14	5.93	3.91	4.59	9.57	3.38
J2	1.94	8.10	1.92	7.77	4.52	6.07	5.94	10.51	10.35	3.80
J3	3.56	9.59	5.37	8.21	6.69	7.77	7.04	9.38	8.19	2.06
J4	4.24	8.80	3.98	5.89	5.45	4.77	6.03	12.68	7.64	1.77
J5	5.80	9.73	2.08	4.47	4.58	5.47	4.45	13.55	8.90	1.88
J6	7.65	10.27	7.58	6.14	4.57	5.90	4.65	7.54	7.85	1.45
J7	7.36	2.49	6.74	3.37	2.07	7.16	2.06	2.21	4.50	3.43
J8	4.40	7.31	4.26	3.99	6.13	5.70	5.99	5.84	6.48	5.33
J9	5.67	3.66	6.20	1.51	6.18	6.97	6.40	2.55	3.62	5.22
J10	3.55	6.03	3.45	3.30	7.34	4.72	6.65	5.95	3.33	5.20
Mean	4.69	6.79	4.61	5.21	5.17	6.05	5.31	7.48	7.04	3.35
f(x2)	3.24	8.72	3.19	4.45	2.08	0.89	2.10	14.22	5.54	2.10

Table 11
Optimal make span values after Iteration 1.

X-New final											
	M/c 1	M/c 2	M/c 3	M/c 4	M/c 5	M/c 6	M/c 7	M/c 8	M/c 9	M/c 10	f(x)
J1	2.71	1.92	4.75	8.91	4.14	5.93	3.57	4.59	9.57	3.38	49.47
J2	2.67	8.10	2.58	3.91	4.52	6.07	7.47	10.51	10.35	3.80	70.65
J3	1.50	5.08	6.23	3.27	6.50	1.09	1.25	7.20	2.58	2.03	111.48
J4	3.19	8.99	3.99	3.96	6.52	2.69	6.24	2.52	2.94	1.44	147.60
J5	3.38	9.33	1.64	1.58	1.13	2.65	2.00	17.99	4.01	1.85	190.47
J6	5.79	15.38	6.02	3.47	6.66	4.78	2.80	10.18	5.31	1.74	235.93
J7	4.66	6.75	2.50	1.82	3.00	5.42	2.76	8.89	6.06	1.81	298.12
J8	8.47	9.87	2.10	1.12	5.64	5.34	2.42	4.90	4.74	5.18	345.17
J9	7.21	6.47	6.83	4.36	5.46	5.81	1.23	1.78	6.34	4.70	394.49
J10	7.16	2.15	6.57	1.45	6.04	5.69	1.62	3.97	3.48	5.10	445.08

Table 12
Optimal machine load values after Iteration 1.

X-New final										
	M/c 1	M/c 2	M/c 3	M/c 4	M/c 5	M/c 6	M/c 7	M/c 8	M/c 9	M/c 10
J1	2.72	1.92	4.57	7.43	4.14	5.93	3.91	4.59	9.57	3.38
J2	1.94	8.10	1.92	7.77	4.52	6.07	5.94	10.51	10.35	3.80
J3	3.56	9.59	5.37	8.21	6.69	7.77	7.04	9.38	8.19	2.06
J4	4.24	8.80	3.98	5.89	5.45	4.77	6.03	12.68	7.64	1.77
J5	5.80	9.73	2.08	4.47	4.58	5.47	4.45	13.55	8.90	1.88
J6	7.65	10.27	7.58	6.14	4.57	5.90	4.65	7.54	7.85	1.45
J7	7.36	2.49	6.74	3.37	2.07	7.16	2.06	2.21	4.50	3.43
J8	4.40	7.31	4.26	3.99	6.13	5.70	5.99	5.84	6.48	5.33
J9	5.67	3.66	6.20	1.51	6.18	6.97	6.40	2.55	3.62	5.22
J10	3.55	6.03	3.45	3.30	7.34	4.72	6.65	5.95	3.33	5.20
Mean	4.69	6.79	4.61	5.21	5.17	6.05	5.31	7.48	7.04	3.35
f(x2)	3.24	8.72	3.19	4.45	2.08	0.89	2.10	14.22	5.54	2.10

their corresponding means for every group.

Step 4 *Teaching Phase*:

In TLBO the student learners from the teacher of their respective classes but in real life, the student not only learns from the class teacher but also learns from discussing with his classmates and also from the tutorial teachers so there is a modification in the formula of to calculate X_{new}. The mean of the particular group is increased by the classroom teacher and tutorial teacher. So the teacher phase formula consists of two parts which is shown in the Eqs. (11)–(13).

For s= 1,2,...... till number of the group
For j=1 to number of design variables

$$X_j'^k = (X^k + DM_{s,j}) + r_i(X_j^h - X_j^k) \text{ if } f(X^h) < f(X^k) \; h \neq k \tag{11}$$

or

$$X_j'^k = (X^k + DM_{s,j}) + r_i(X_j^k - X_j^h) \text{ if } f(X^k) < f(X^h) \; h \neq k \tag{12}$$

$$DM_{s,j} = r_i(X_j^s - T_F * Mean_{s,j}) \tag{13}$$

X_j^s *is* the grade of the teacher.
$DM_{s,j}$ *is* the difference mean.
$Mean_{s,j}$ is the mean grade of the group.

The $f(X_{new})$ which has been generated for the each learner is compared with the f(X) value and whichever values are lesser are retained and are carried to the next phase of MOTLBO. The calculated DM values

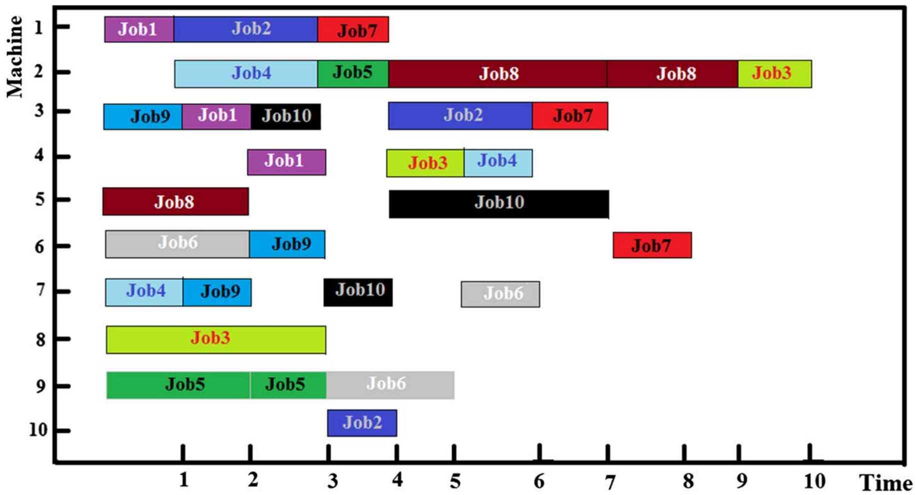

Fig. 2. GANTT chart for scheduling of jobs after Iteration 1.

Table 13
Archived population for the Iteration 1 values of make span from the m/c workload values.

X-New final of make span											
	M/c 1	M/c 2	M/c 3	M/c 4	M/c 5	M/c 6	M/c 7	M/c 8	M/c 9	M/c 10	f(x)
J1	2.71	1.92	4.75	8.91	4.14	5.93	3.57	4.59	9.57	3.38	49.47
J2	2.67	8.10	2.58	3.91	4.52	6.07	7.47	10.51	10.35	3.80	70.65
J3	1.50	5.08	6.23	3.27	6.50	1.09	1.25	7.20	2.58	2.03	111.48
J4	3.19	8.99	3.99	3.96	6.52	2.69	6.24	2.52	2.94	1.44	147.60
J5	3.38	9.33	1.64	1.58	1.13	2.65	2.00	17.99	4.01	1.85	190.47
J6	5.79	15.38	6.02	3.47	6.66	4.78	2.80	10.18	5.31	1.74	235.93
J7	4.66	6.75	2.50	1.82	3.00	5.42	2.76	8.89	6.06	1.81	298.12
J8	8.47	9.87	2.10	1.12	5.64	5.34	2.42	4.90	4.74	5.18	345.17
J9	7.21	6.47	6.83	4.36	5.46	5.81	1.23	1.78	6.34	4.70	394.49
J10	7.16	2.15	6.57	1.45	6.04	5.69	1.62	3.97	3.48	5.10	445.08
Mean	4.67	7.40	4.32	3.39	4.96	4.55	3.13	7.25	5.54	3.10	
f(x2)	5.01	14.08	3.66	4.67	2.89	2.76	4.01	21.28	6.31	2.05	

Table 14
Archived population for the Iteration 1 values of make work load from the m/c makespan values.

X-New final of workload											
	M/c 1	M/c 2	M/c 3	M/c 4	M/c 5	M/c 6	M/c 7	M/c 8	M/c 9	M/c 10	f(x)
J1	2.72	1.92	4.57	7.43	4.14	5.93	3.91	4.56	2.64	3.38	41.20
J2	1.94	8.10	1.92	7.77	4.52	6.07	5.94	12.38	8.93	3.80	71.00
J3	3.56	9.59	5.37	8.21	6.69	7.77	7.04	9.33	0.74	2.06	107.72
J4	4.24	8.80	3.98	5.89	5.45	4.77	6.03	4.55	2.91	1.77	164.69
J5	5.80	9.73	2.08	4.47	4.58	5.47	4.45	13.43	5.09	1.88	213.20
J6	7.65	10.27	7.58	6.14	4.57	5.90	4.65	5.32	2.17	1.45	269.73
J7	7.36	2.49	6.74	3.37	2.07	7.16	2.06	2.25	1.49	3.43	327.40
J8	4.40	7.31	4.26	3.99	6.13	5.70	5.99	4.08	3.06	5.33	367.72
J9	5.67	3.66	6.20	1.51	6.18	6.97	6.40	2.52	0.72	5.22	417.85
J10	3.55	6.03	3.45	3.30	7.34	4.72	6.65	5.99	3.31	5.20	462.89
Mean	4.69	6.79	4.61	5.21	5.17	6.05	5.31	6.44	3.11	3.35	
f(x2)	3.24	8.72	3.19	4.45	2.08	0.89	2.10	13.96	5.29	2.10	

for the above problem are shown in Tables 7 and 8 and X_{new} values for the same problem is shown in Tables 9 and 10.

Step 5: *Learner phase*

In learner phase of TLBO, the student learns from his peer and tries to improve his performance, but in real life, the student learns from his peer not only from his peers but also from improves his performance by self-motivation by improving his knowledge self-studying. So in MOTLBO algorithm, there has been a modification in the search mechanism so that the above condition is satisfied. The mathematical formula of learner phase of MOTLBO is modified represented in the Eqs. (13)–(15).

For j= 1to number of design variable

$$X'^{p}_{j,i} = [X^{p}_{j,i} + r_i(X^{p}_{j,i} - X^{q}_{j,i})] + [r_i(X^{S}_{j,i} - E_F * X^{p}_{j,i})],\ \text{If } f(X^{p}) < f(X^{q}) \quad (13)$$

or

$$X'^{p}_{j,i} = [X^{p}_{j,i} + r_i(X^{q}_{j,i} - X^{p}_{j,i})] + [r_i(X^{S}_{j,i} - E_F * X^{p}_{j,i})],\ \text{If } f(X^{q}) < f(X^{p}) \quad (14)$$

$p \neq q$ and p,q,s $\in$k,

X^{S}_{j} *is the* grade of the teacher associated with group 's' in 'j' subject

Table 15
Input parameters values of the proposed algorithms.

Input parameters of the considered algorithms	Parameters value
CNSGA-II	
Population size	200
Total number of Iterations	150 to 300
Probability of cross over (Pc)	0.65 to 0.9
Probability of mutation (Pm)	0.01 to 0.1
MOPSO	
Cognitive factor *(c1)*	0.5 to 2
Social factor *(c2)*	0.5 to 2
Swarm size *(N)*or Population Size	200
Number of iterations *(K)*or Total number of generations	150 to 300
MOTLBO	
Population size	200
Total number of generations	150 to 300
Teacher factor	1 or 2
Probability of cross over (Pc)	0.65 to 0.9
Probability of mutation (Pm)	0.001 to 0.1

Table 16
Experimental conditions of different attributes with their values.

Type	Attributes	Values
Shop floor	Size	6–15
	Machine breakdown level	0.025
	MTTR	5
	MTBF, MTTR	Exponential Distribution
	Key machine	Maximum 1 with random
Performance measures	Minimization of Makespan	
	Minimization of work load on most loaded machine	

$$E_F = \text{Round}\,(1 + r_i) \tag{15}$$

E_F is the exploring factor of the learner whose value can either be 1 or 2. The E_F is the self-motivation factor of the learner that means the student can learner can learn everything from self-motivation or else nothing from it. The Table 11 and 12 represents the final X_{new} value of learner phase of the first generation of the above problem. GANTT chart for scheduling of jobs is presented in Fig. 2.

Step 6 *External Archive:*

The main function of the external archive is to record of non-dominated vectors found along the search process available in each generation. This algorithm uses a ε-dominance method to store the data of each generation stored in the fixed sized external archeries to store best solution that has been generated in that particular generation. The detailed study about ε-dominance is explained in [26].

Based on the number of objective functions of the problem archive space will be located in such a way that the dimension of the space is equal to the number of the objective function. Depending on the number of objective functions that is two, three or more than three the dimensional area of the space is divided into square, cubic or hypercubic shapes. In a box if there is only one solution the value is stored, if there is more than one solution in a particular box the values which can dominate the remaining values is kept and remaining values will be removed, then the box have particular values is checked with remaining values and try to find weather the boxes possess the property of non-dominance or not. If the values present in the boxes are dominated by the remaining values then such values will be removed. The problem is solved under MOTLBO are solved by grid-based approach for achieving the values.

Once the values that have been received after teacher phase and learner phase is continued for a certain number of 100 to 150 iterate or until there is no significance improvement in the values or the values keeps on increasing. At the end of the algorithm the values that have been stored in the external archive is given as output at the end of the iteration.

The Tables 13 and 14 represents the output received from makespan and m/c workload respectively. For each table the other objective function values have been calculated. These two tables actually explains what happens in the external archive where each table is better in the objective function it is calculated from and deviating away from the satisfying from other objective function. Once these values are found out and are verified and the one which gets dominated is eliminated keeping the other values in the box.

4. Experimental evaluation

To solve the proposed problem with the presented model different test problem instances are considered from Brandimarte [4]. For example, we have shown the 10- jobs, 10-machines FJSSP case as an instance in Table 1. The presented problem is complex, multi-objective and NP-hard in nature, therefore, multi-objective evolutionary algorithm approach is considered as a root to solve the problem. Here, with proposed hybrid MOTLBO algorithm the presented test instances are solved. Consequently the performance of the proposed algorithm is compared with those of the obtained NSGA-II and MOPSO. In Table. 15 the proposed and considered algorithms are their parameter values are compared.

In this study, we consider machines break down as a real-time event and its effect on system performance is examined. To understand the failure rate, we assume all the machines have the same mean time to repair (*MTTR*) and mean time between failure (*MTBF*) where $Avg = MTTR/(MTBF+MTTR)$, it denotes the percentage of machines those are broken and have failures. For example, if $MTTR = 5$ time units and $MTBF = 45$ time units then $Avg = 0.5$, this indicates on an average 45

Table 17
Makespan and total machine load variation of different without machine break down scenarios of different algorithms.

Brandimarte benchmark Instances				Makespan			Total Machine load variation		
Scenarios	number of jobs	number of machines	LB	Proposed MOTLBO algorithm	MOPSO	NSGA-II	Proposed algorithm	MOPSO	NSGA-II
MK01	10	6	36	40	40	40	44	49	52
MK02	**10**	**6**	**24**	**24**	**28**	**29**	**52**	**67**	**74**
MK03	15	8	204	287	300	321	432	474	488
MK04	**15**	**8**	**48**	**97**	**106**	**112**	**200**	**225**	**246**
MK05	15	4	168	186	203	205	413	448	471
MK06	**10**	**15**	**33**	**58**	**65**	**72**	**176**	**187**	**192**
MK07	20	5	133	140	152	167	298	324	345
MK08	**20**	**10**	**523**	**530**	**545**	**556**	**786**	**792**	**812**
MK09	20	10	299	315	340	358	564	574	589
MK10	20	15	165	212	228	254	310	345	356

Table 18
Makespan and total machine load variation of different instances with machine break down scenarios of different algorithms.

Scenarios	Number of jobs	Number of machines	LB	Proposed MOTLBO algorithm	MOPSO	NSGA-II	Proposed algorithm	MOPSO	NSGA-II
MK01	10	6	36	56	62	69	144	154	175
MK02	**10**	**6**	**24**	**39**	**48**	**52**	**85**	**98**	**146**
MK03	15	8	204	332	374	397	651	724	906
MK04	**15**	**8**	**48**	**123**	**136**	**148**	**322**	**386**	**554**
MK05	15	4	168	227	265	271	964	1211	1456
MK06	**10**	**15**	**33**	**83**	**94**	**112**	**310**	**417**	**621**
MK07	20	5	133	172	246	263	677	748	873
MK08	**20**	**10**	**523**	**574**	**623**	**631**	**1024**	**1347**	**1546**
MK09	20	10	299	384	392	427	816	920	1144
MK10	20	15	165	242	275	280	624	754	837

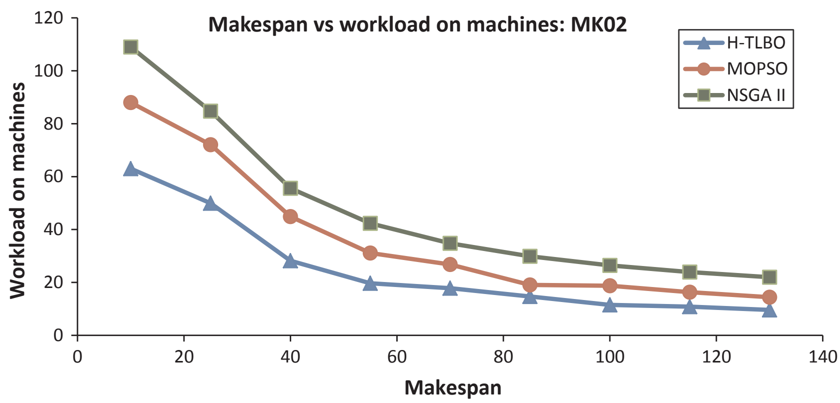

Fig. 3a. Pareto optimal curves of Instance 2 for three algorithms.

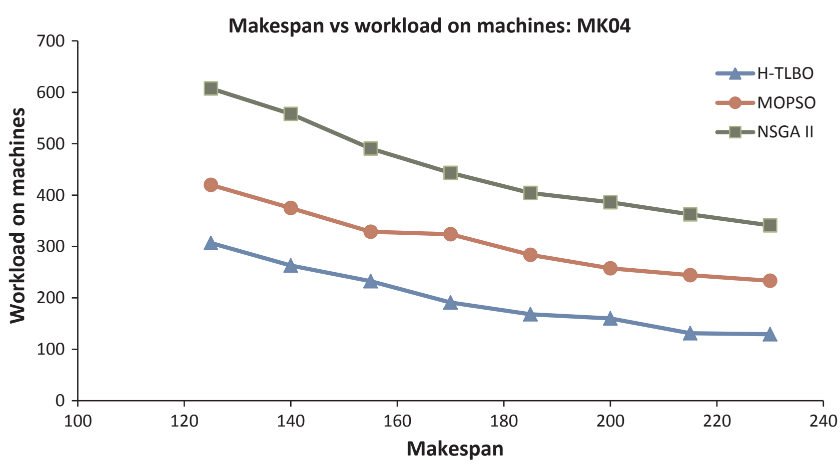

Fig. 3b. Pareto optimal curves of Instance 4 for three algorithms.

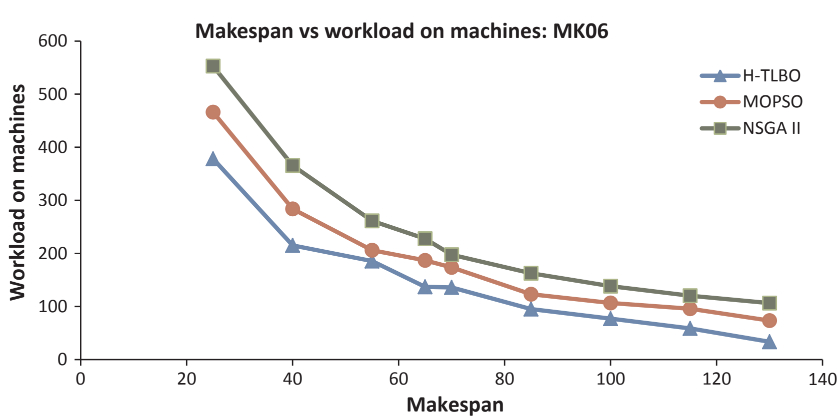

Fig. 3c. Pareto optimal curves of Instance 6 for three algorithms.

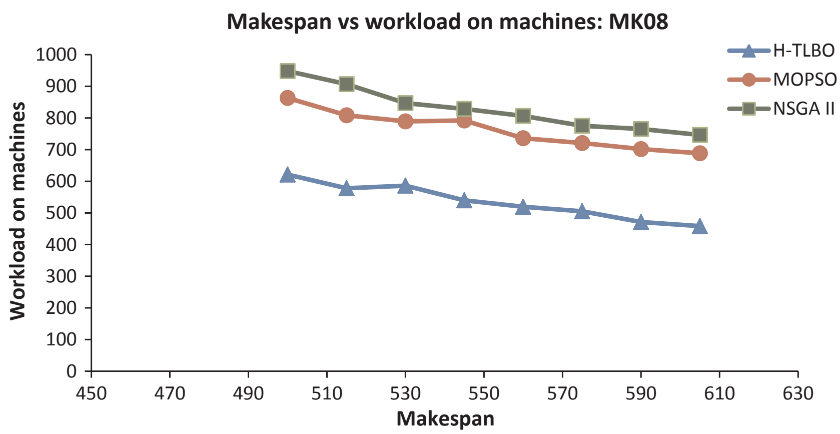

Fig. 3d. Pareto optimal curves of Instance 8 for three algorithms.

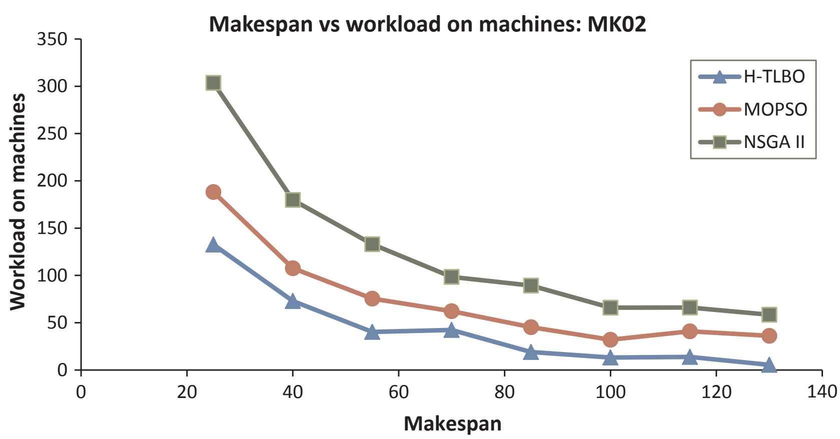

Fig. 4a. Pareto optimal curves of Instance 2 for three algorithms with consideration of machines break down.

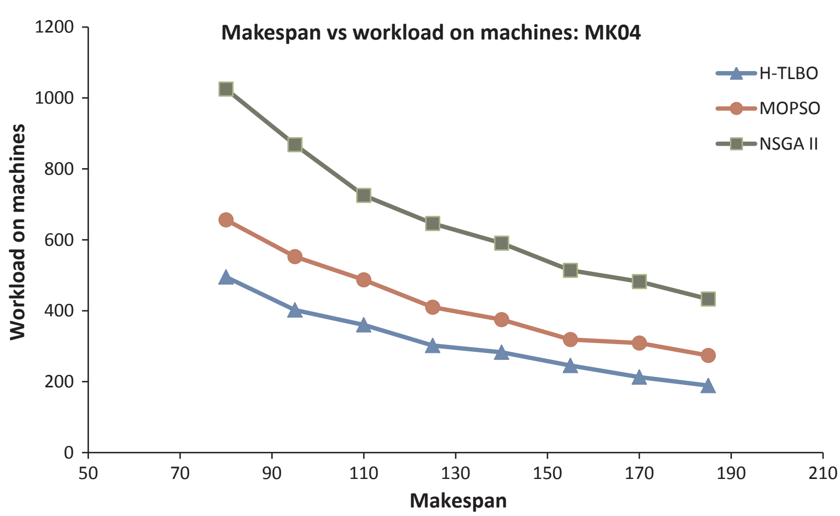

Fig. 4b. Pareto optimal curves of Instance 4 for three algorithms with consideration of machines break down.

time units the machine is available and the machine breaks down with the mean of 5 time units. The attributes and its values are shown in Table. 16.

Among all parameters of the presented algorithms, population size and termination criteria are most common and important parameters to investigate. In this paper, we have considered difference values for termination criteria ranging from 150 to 300 by fixing population size as constant with 200. To solve the proposed problem, each instance is run for ten times by set the termination criteria for each algorithm. The average of ten runs is taken as best solutions for each instance of two different cases is shown in Tables 17 and 18. The algorithms were coded in MATLAB software and the problem is executed on a computer with Intel® Core™2 Duo CPU T7250 @2.00 GHz, 1.99 GB of RAM.

5. Results and discussion

In this paper, the proposed systems performance and its behaviour when real-time event occurs is examined with considered conflicting objectives such as minimization of makespan and machine load variation. In this research work, a multi-objective evolutionary algorithmic approach is considered due to their nature to solve the conflicting objectives simultaneously and help to increase the solutions diversity with better computational time. Since these algorithms use Pareto optimal fronts to represent the optimal solutions, the set of solutions at the last

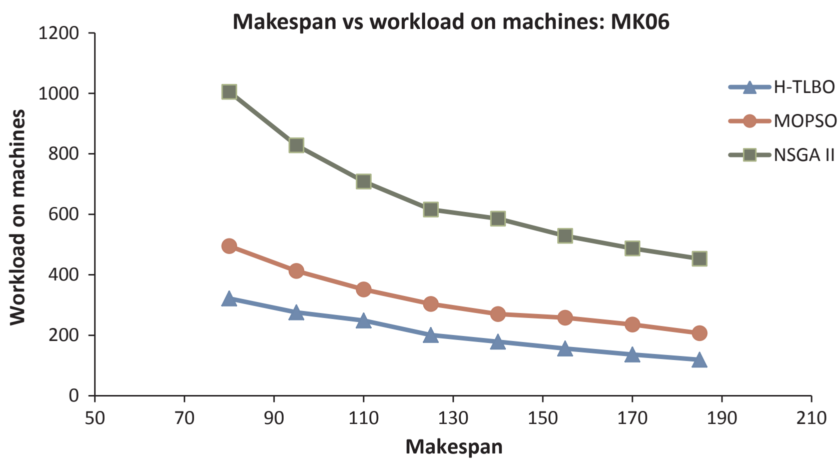

Fig. 4c. Pareto optimal curves of Instance 6 for three algorithms with consideration of machines break down.

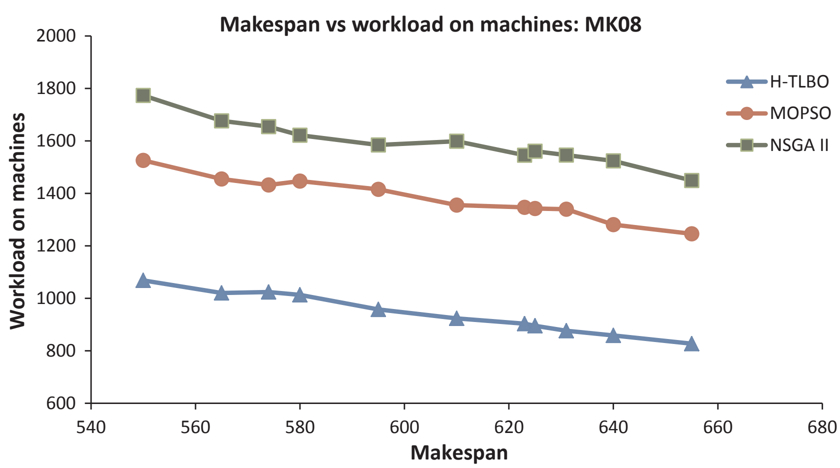

Fig. 4d. Pareto optimal curves of Instance 8 for three algorithms with consideration of machines break down.

Table 19
Comparison of results with machines break down and without machines break down with different algorithms.

Brandimarte benchmark Instances				Without machines break down						With machines break down					
				Makespan			Total Machine load variation			Makespan			Total Machine load variation		
Scenarios	Number of jobs	Number of machines	LB	Proposed MOTLBO algorithm	MOPSO	NSGA-II	Proposed algorithm	MOPSO	NSGA-II	Proposed MOTLBO algorithm	MOPSO	NSGA-II	Proposed MOTLBO algorithm	MOPSO	NSGA-II
MK02	10	6	24	24	28	29	52	67	74	39	48	52	85	98	146
MK04	15	8	48	97	106	112	200	225	246	123	136	148	322	386	554
MK06	10	15	33	58	65	72	176	187	192	83	94	112	310	417	621
MK08	20	10	523	530	545	556	786	792	812	574	623	631	1024	1347	1546

iteration is considered as the Pareto optimal solutions. Ten different scenarios from Brandimarte [4], has been taken to conduct the experiments with developed hybrid MOTLBO algorithm. Subsequently, comparisons with two existing multi-objective-based algorithms including conventional NSGA-II, MOPSO is made to find the proposed algorithm effectiveness.

5.1. Experimentation results without considering machines breakdown

Table 4 and 5 indicates that most of the obtained solutions for considered objectives are dominated by solutions obtained from other algorithms. Figs. 3–d shows the non-dominated solutions of the tested algorithms for randomly chosen instances MK02, MK04, MK06, and MK08 out of ten instances in which machines break down is not considered. The X-axis in Figs. 3a–d indicates make-span as a performance measure and Y-axis indicates machine load variation as a performance measure and the generated Pareto optimal curves for three different algorithms are shown with three different symbols. The plots illustrate the convergence and divergence solutions of the three algorithms which depict the performance of the system. The extreme curve indicates high convergence and diversity with optimal Pareto solutions.

5.2. Experimental results with consideration of machines breakdown

In this section, the performance of the systems is examined when machines break down occur and also how the proposed algorithm improves the performance measures is detailed. We use similar instances and generated machines breakdown randomly. As indicated in Table 3, the related parameters for machine breakdown such as MTBF and MTTR are assumed with repair time between failures of machines as an exponential distribution. Figs. 4a–d shows the non-dominated solutions of the tested algorithms for randomly chosen instances MK02, MK04, MK06, and MK08 out of ten instances. X-axis and Y-axis in the below-mentioned figures show the considered performance measures and Pareto curves show the performance of the algorithms.

From the results it is suggested that the proposed MOTLBO is effective compared to other algorithms where it produces well spread non-dominated solutions for almost all the instances. As indicated in Table 19, the scheduling efficiency is affected when machines breakdown is taken into consideration and the proposed algorithm consistently improved the performance of the system.

6. Conclusion and future work

In this paper, several contributions for literature review with the proposed FJSSP has been made. Primarily, a multi-objective based FJSSP has been proposed with machines break down as a real time event. Secondarily, we developed a mathematical model with objective functions as minimization of total completion time and machine load variations to test the system performance. The proposed problem is a multi-objective and combinatorial optimization in nature where finding the compromising solution for the conflicting objectives by using conventional optimization techniques is very hard. In this regard, to solve this problem as a third contribution, we proposed a hybrid multi-objective based evolutionary algorithm i.e., MOTLBO to find the optimal solutions. With different benchmark data sets available from the literature the problem has been solved for two different scenarios i.e., with and without machine breakdown with various instances. Computational experiments have shown that the proposed MOTLBO algorithm outperforms the other algorithms and is able to give useful near optimum results for the above mentioned scenarios. Further, with already established MOPSO and CNSGA-II algorithms the proposed algorithm is compared and from the results it has been proved that the proposed algorithm shows its best over other algorithms in almost all the test problems. Moreover, the proposed algorithm eliminates the limitation of the variations in the results shown by the other algorithms by giving close results for both the cases. Different examinations are conducted to test the robustness of the proposed system when the machines break down occurs.

The future research work may include incorporation of more real-time events to make the considered system as more practical FJJSP. The proposed approach may be applied to improve the Pareto optimal fronts for generating best compromising non-dominated optimal solutions for many objectives based FJSSP.

References

[1] P. Brucker, R. Schile, Job shop scheduling with multi-purpose machines, J. Comput. 45 (1990) 369–375.
[2] M.R. Garey, D.S. Johnson, R. Sethi, The complexity of flow shop and job shop scheduling, Mathemat. Operat. Res. 1 (1976) 117–129.
[3] G. Zhang, L. Gao, Y. Shi, An effective genetic algorithm for the flexible job shop scheduling problem, Expert Syst. Appl. 38 (2011) 3563–3573.
[4] P. Brandimarte, Routing and scheduling in a flexible job shop by Tabu search Annals, J. Operat. Res. 41 (1993) 157–183.
[5] J. Hurink, B. Jurisch, M. Thole, Tabu search for the job shop scheduling problem-with multi-purpose machines, J. OR Spect. 15 (1994) 205–215.
[6] N.M. Najid, S. Dauzere-Peres, A. Zaidat, A modified simulated annealing methodfor flexible job shop scheduling problem, IEEE international conference on system, man and cybernetics, (2002) 5.
[7] J.B. Yang, GA-based discrete dynamic programming approach for scheduling in FMS environments, IEEE Transact. Sys., Man Cybernetics, Part B 31 (2001) 824–835.
[8] J.X. Tay, N.B. Ho, Evolving dispatching rules using genetic programming for solving multi-objective flexible job shop problems, J. Comput. Indust. Eng. 54 (2008) 453–473.
[9] L. Wang, S. Wang, Y. Xu, G. Zhou, M. Liu, A bi-population based estimation of-distribution algorithm for the flexible job-shop scheduling problem, J. Comput. Indust. Eng. 64 (2011) 917–926.
[10] Q. Zhang, H. Manier, M.A. Manier, A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constraints and bounded processing times, J. Comput. Operat. Res. 39 (2011) 1713–1723.
[11] W. Xia, Z. Wu, An effective hybrid optimization approach for multi-objective flexible job shop scheduling problems, J. Comput. Indust. Eng. 48 (2005) 409–425.
[12] G.H. Zhang, X.Y. Shao, P.G. Li, L. Gao, A genetic algorithm and tabu search for multiobjective flexible job shop scheduling problems. IEEE International Conference on Computing, J. Control Indust. Eng. 1 (2010) 251–254.
[13] A. Bagheri, M. Zandieh, Bi-criteria flexible job shop scheduling with sequence-dependent setup times—variable neighbourhood search approach, J. Manufact. Sys. 30 (2011) 8–15.
[14] E.L. Ulungu, J. Teghem, P.H. Fortemps, D. Tuyttens, MOSA method: a tool for solving MOCO problems, J. Multi-Criteria Decis. Analys. 8 (1999) 221–236.
[15] J. Kennedy, R.C. Eberhart, Swarm Intelligence, San Mateo, CA, Morgan Kaufmann, 2001.
[16] W. Xia, Z. Wu, An effective hybrid optimization approach for multi objective flexible job-shop scheduling problems, J. Comput. Indust. Eng. 48 (2005) 409–425.
[17] G. Zhang, X. Shao, P. Li, L. Gao, An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem, J. Comput. Indust. Eng. 56 (2009) 1309–1318.
[18] D. Lei, Pareto archive particle swarm optimization for multi-objective fuzzy job shop scheduling problems, Internat. J. Adv. Manufact. Tech. 37 (2008) 157–165.
[19] R.V. Rao, V. Savsani, D. Vakharia, Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, J. Comput. Aid. Des. 43 (2011) 303–315.
[20] R.V. Rao, V. Patel, Multi-objective optimization of heat exchangers using a modified teaching–learning-based optimization algorithm, J. Appl. Mathemat. Model 37 (2012) 1147–1162.
[21] R.V. Rao, V. Patel, Multi-objective optimization of two stage thermoelectric cooler using a modified teaching–learning-based optimization algorithm, J. Eng. Appl. Artif. Intelligence 26 (2013) 430–445.
[22] V. Toğan, Design of planar steel frames using teaching–learning based optimization, J. Eng. Struct. 34 (2012) 225–232.
[23] K.R. Krishnanand, B.K. Panigrahi, P.K. Rout, A. Mohapatra, Application of multi-objective teaching–learning-based algorithm to an economic load dispatch problem with incommensurable objectives, in: B.K. Panigrahi (Ed.), Swarm, Evolutionary, and Memetic Computing, Springer Verlag, Berlin, 2011, pp. 697–705.
[24] R.V. Rao, V. Kalyankar, Parameter optimization of modern machining processes using teaching–learning-based optimization algorithm, J. Eng. Appl. Artif. Intelligen. 26 (2013) 524–531.
[25] D.E. Goldberg, B. Korb, K. Deb, Messy genetic algorithms: Motivation, analysis, and first results, J. Complex Syst. 3 (5) (1989) 493–530.
[26] K. Deb, H.G. Beyer, Self-adaptive genetic algorithms with simulated binary cross-over, J. Evolution. Comput. 9 (2) (2001) 197–221.
[27] F. Khoukhi, J. Boukachour, A. Alaoui, The “Dual-Ants Colony”: A novel hybrid approach for the flexible job shop scheduling problem with preventive maintenance, Comput. Ind. Eng. 106 (5) (2017) 236–255.
[28] C. Lu, X. Li, L. Gao, W. Liao, J. Yi, An effective multi-objective discrete virus optimization algorithm for the flexible job shop scheduling problem with controllable processing times, Comput. Ind. Eng. 104 (8) (2017) 156–174.
[29] A. Javid, P. Hooshangi, Integrating employee timetabling with scheduling of machines and transporters in a job shop environment: a mathematical formulation and an Anarchic society optimization algorithm, Comput. Oper. Res. 84 (3) (2017) 73–91.
[30] H. Nouri, B. Driss, K. Ghédira, Simultaneous scheduling of machines and transport robots in flexible job shop environment using hybrid metaheuristics based on clustered holonic multiagent model, Comput. Ind. Eng. 102 (2016) 488–501.
[31] V. Kaplanoğlu, An object-oriented approach for multi-objective flexible job shop scheduling problem, Expert Syst. Appl. 45 (2016) 71–84.
[32] V.K. Manupati, S. Deo, N. Cheikhrouhou, M.K. Tiwari, Optimal process plan selection in networked based manufacturing using game-theoretic approach, Int. J. Prod. Res. 50 (18) (2012) 5239–5258.
[33] M. Yazdani, M. Amiri, M. Zandieh, Flexible job-shop scheduling with parallel variable neighborhood search algorithm, Expert Syst. Appl. 37 (1) (2010) 678–687.
[34] A.S. Fleming, D. Ruble, H. Krieger, P.Y. Wong, Hormonal and experiential correlates of maternal responsiveness during pregnancy and the puerperium in human mothers, Horm. Behav. 31 (2) (1997) 145–158.
[35] E. Zitzler, S. Künzli, Indicator-based selection in multiobjective search, International Conference on Parallel Problem Solving from Nature, Springer, Berlin, Heidelberg, 2004, pp. 832–842.