The International Journal of Advanced Manufacturing Technology (2019) 100:1419–1432
https://doi.org/10.1007/s00170-018-2805-0

ORIGINAL ARTICLE

Two-stage teaching-learning-based optimization method for flexible job-shop scheduling under machine breakdown

Raviteja Buddala [1] · Siba Sankar Mahapatra [1]

Received: 19 September 2017 / Accepted: 28 September 2018 / Published online: 3 October 2018

Abstract

In the real-world situations, uncertain events commonly occur and cause disruption of normal scheduled activities. Consideration of uncertain events during the scheduling process helps the organizations to make strategies for handling the uncertainties in an effective manner. Therefore, in the present paper, unexpected machine breakdowns have been considered during scheduling of jobs in a flexible job-shop environment. The objective is to obtain lowest possible makespan such that robust and stable schedules are produced even if an unexpected machine breakdown occurs. The robust and stable schedules may help to decrease the costs associated with unexpected machine failures. The present work uses a two-stage teaching-learning-based optimization (2S-TLBO) method to solve flexible job-shop scheduling problem (FJSP) under machine breakdown. In the first stage, the primary objective of makespan is optimized without considering any machine breakdown. In the second stage, a bi-objective function considering robustness and stability of the schedule is optimized under uncertainty of machine breakdowns. In order to incorporate the machine breakdown data to basic FJSP, a non-idle time insertion technique is used. In order to generate effective robust and stable predictive FJSP schedules, a rescheduling technique called modified affected operations rescheduling (mAOR) is used. The Kacem's and Brandimarte's benchmark problems have been solved and compared with other algorithms available in the literature. Results indicate that TLBO outperforms other algorithms by generating superior robust and stable predictive schedules. Statistical analysis is carried out to test the significance difference of the results obtained by TLBO with other algorithms.

Keywords Flexible job-shop scheduling · Machine breakdown · Makespan · Meta-heuristics · Robustness · Stability · Teaching-learning-based optimization

1 Introduction

In the last few decades, job-shop scheduling problem (JSP) has been studied extensively. The flexible job-shop scheduling problem (FJSP) is an extension of basic JSP where more than one processor is available at each facility. In the last decade, FJSP has drawn good attention from research community. Most of the studies have addressed the FJSP with makespan as the objective [1–13]. For a scheduling problem, the makespan is defined as the completion time of the last job to leave the production system [14]. Many studies have solved FJSP with multi-objective criteria considering makespan, total work load, and critical work load simultaneously [15–24]. While addressing above objectives, strict assumption is made that no machine failure occurs during the processing of the jobs and the processing times of operations are deterministic and known at priori. Both the assumptions make it difficult to generate a good schedule for a real-world FJSP which is subjected to various uncertainties like machine breakdown and variable processing times. A shop floor may also experience other uncertainties like sudden job arrivals or job cancellations [25], operator illness, and due date changes. However, the probability of execution of a schedule as per plan in a real-world shop floor is very low due to uncertainties that arise in a shop floor [26].

It is clear from the past literature that sufficient studies have not been devoted to tackle uncertainties in FJSP. The uncertainties involved in a job-shop environment can be categorized into two types such as (i) resource related and (ii) job

✉ Raviteja Buddala
raviteja317@gmail.com

Siba Sankar Mahapatra
mahapatrass2003@gmail.com

[1] Department of Mechanical Engineering, National Institute of Technology Rourkela, Rourkela, India

related [27]. Shortage or delay in the arrival of materials, operator illness, loading limits, and unavailability of machines or failure of tools are labelled under resource related uncertainties. Changes in due dates, unexpected arrival of new jobs or cancellation of existing jobs, uncertainties in processing times, early arrival of jobs, or delay in arrival of jobs are treated as job-related uncertainties. All these uncertainties ought to be taken into account if it is intended to generate a schedule considering uncertainties [28]. Therefore, the algorithms and procedures that have been applied to solve deterministic scheduling problems cannot be applied to scheduling problems with uncertainty [29].

In addition to the existing performance measures like makespan, tardiness, and flow time, two more performance measures like robustness and stability come into picture in case of flexible job-shop scheduling considering machine breakdown uncertainty. Liu et al. [30] and Liu et al. [31] have defined robust schedule as a schedule that is insensitive to disruptions, i.e., a schedule may degrade its performance to a very small degree under interruptions. Bidot et al. [32] have defined robust schedule as the quality of executed schedule close to the quality of predicted schedule. Xiong et al. [33] have expressed robustness as the difference between the actual makespan and the deterministic makespan of predicted schedule. Wu et al. [34] have defined stable schedule as the one which has less deviation in terms of time or sequence of operations between the realized and predicted schedule. Goren and Sabuncuoglu [35] have defined stable schedule as a schedule which do not deviate much from the initial schedule, and stability is expressed as the difference between the completion times of realized and initial schedule. Studies have been made on different robustness and stability measures by Jensen [36], Al-Hinai and ElMekkawy [37], and He et al. [29] for FJSP using the measures that were previously applied to JSP.

Scheduling under uncertainty has drawn the interest of research community in the recent years [38]. JSP itself is an NP hard (non-deterministic polynomial time) problem, and FJSP which is an extension of JSP is a much more difficult problem to solve. In order to address the uncertainties, data related to uncertainties need to be incorporated into the existing problem. This makes the FJSP problem much more difficult to formulate and solve. In order to solve these problems, heuristic and meta-heuristic techniques have been extensively used [39]. In case of machine breakdown, FJSP becomes more complex to solve as compared to its deterministic case [29]. Chaari et al. [28] have proposed a classification of the FJSP under machine breakdown problem into three categories such as (i) proactive methods, (ii) reactive methods, and (iii) hybrid methods. Based on the previous statistical knowledge of uncertainty in FJSP environment, proactive or predictive methods (offline techniques) produce a predictive schedule with an aim to generate a schedule of good average performance. A schedule that is predetermined or precomputed is called a predictive schedule or preschedule which is executed until there is a machine breakdown in the shop floor. Next, rescheduling methods are applied to tackle the machine breakdown. In order to absorb the adverse effect of breakdown in scheduling, two approaches are used (i) supply of new external resource and (ii) an extension of processing time of the operation due to repair time [40]. A scheduling approach that produces a predictive schedule to reduce the adverse effect of machine breakdown is proposed by Al-Hinai and ElMekkawy [37] using hybrid genetic algorithm (GA). In addition, they have proposed three stability measures for FJSP. A modified hybrid GA is proposed by Al-Hinai and ElMekkawy [39] to solve FJSP using uniform distribution to represent the processing times of operations. Dalfard and Mohammadi [41] have proposed mathematical foundation and two meta-heuristics such as hybrid GA and simulated annealing (SA) to solve FJSP with parallel machines with an aim to minimize maintenance cost. Out of all the uncertainties in a FJSP environment, machine breakdown is the most focused uncertainty [29].

In reactive scheduling methods, no previous statistical knowledge of uncertainty is required. Unlike proactive or predictive methods, reactive methods do not generate prior schedules. The schedules are generated on the real-time basis when a machine breakdown occurs. These are also known as online methods (Liu et al. [30] and Liu et al. [31]). With an aim to dispatch jobs on the available machines dynamically with some preassigned priorities, priority dispatching rules are frequently implemented in reactive approach. They also present a method that can react to various disruptions in a dynamic job-shop scheduling based on a multi-agent framework which combines a predictive decision-making with real-time decision-making. Mouelhi-Chibani and Pierreval [42] have proposed a neural network (NN)-based approach which selects the dispatching rules that suits best in real time. Without the use of any prediction, Zbib et al. [43] have used potential fields to allocate jobs dynamically and proposes a reactive approach to solve FJSP.

Recently, hybrid methods, a combination of proactive and reactive methods, are used and known as proactive-reactive methods or predictive-reactive methods [28]. A predictive-reactive method produces a predictable schedule that is prepared reactively for absorbing the disruptions taking uncertain events into account. It aims at reducing the impact of disturbances on the original schedule. There are two steps in these methods. In the first step, an offline deterministic schedule is generated. Therefore, production begins using the predictive schedule and it is adapted online in the second step [44]. Gao et al. [45] propose different heuristics to solve FJSP problem with an aim to insert a newly arrived job in the schedule. When a new job arrives, they use the same heuristic techniques during the rescheduling phase. Based on the real-time manufacturing data, Wu et al. [46] use a reactive method in the rescheduling stage which spontaneously generates a revised schedule when an unexpected disturbance occurs. However, there is a slight difference between proactive-reactive and predictive-reactive methods. In case of proactive-reactive methods, one among many pregenerated schedules is selected and no online rescheduling is done. On the other hand, online rescheduling is done in case of predictive-reactive methods [28].

Machine breakdown is one of the important uncertainties encountered in real practice. It means that a machine is unavailable for processing due to some failure. The same machine can become available for processing only when the required repair work is done. If a schedule is generated assuming that there is no machine breakdown, the planned schedule may get delayed for duration equal to machine repair time. This can cause production delay and the organizations may not deliver the products within the promised due dates affecting the good will of the company. Therefore, firms require schedules which are robust as well as stable. To address these issues, in the present work, a two-stage teaching-learning-based optimization (2S-TLBO) algorithm is proposed. In the first stage, the algorithm optimizes the first objective (makespan) to generate a schedule based on deterministic approach without considering any machine breakdowns. In the second stage, a bi-objective function considering robustness and stability under machine breakdown situation is solved. In order to solve FJSP under machine breakdown condition, there is a need to incorporate machine breakdown data into the basic FJSP problem. For this purpose, a non-idle time insertion technique is used. The inserted non-idle time acts as a buffering time during a machine breakdown without affecting the actual schedule. When a machine breakdown occurs, a rescheduling technique is required in order to generate effective robust and stable predictive FJSP schedules. Without a proper rescheduling technique, the makespan of a disruptive schedule increases leading to poor stability of the schedule. In the present work, a technique called modified affected operations rescheduling (mAOR) proposed by Subramaniam and Raheja [25] is used. A detailed discussion on 2S-TLBO, non-idle time insertion technique, and mAOR rescheduling technique is given in later sections of the paper.

2 Literature review

Li and Cao [47] have studied machine breakdown on scheduling problems like single machine, flow shop, and open shop problems. Kasap et al. [48] have applied priority rule like least processing time (LPT) rule to a single machine scheduling problem to calculate expected makespan with an unreliable machine. Two-stage production scheduling for a flow shop scheduling problem with stochastic machine breakdowns and setup times is studied by Allahverdi [49]. Flow shop scheduling problems with an aim to minimize maximum lateness are studied by Allahverdi [50]. Allahverdi and Mittenthal [51] and Allahverdi [52] have studied two machine flow shop scheduling problems under machine breakdown using heuristic approaches. Alcaide et al. [53] have proposed a new procedure to solve the flow shop scheduling problems under machine breakdown by converting the problem into an equivalent deterministic problem without any breakdown. Alcaide et al. [54] have proposed a heuristic approach to solve open shops under machine breakdown with an aim to minimize makespan. Luh et al. [55] have solved job-shop scheduling problem considering different uncertainties like machine breakdown, variable processing times and due dates. Lei [56] has proposed a GA with random key representation to solve job-shop scheduling problem under random machine breakdown. Ahmadi et al. [57] have proposed evolutionary algorithms to solve multi-objective FJSP under machine breakdown. Park et al. [58] have proposed a genetic programming based hyper-heuristic approach to solve dynamic job-shop scheduling under machine breakdown. Zandieh et al. [59] have proposed an improved imperialist competitive algorithm for condition-based maintenance of FJSP. El Khoukhi et al. [60] have proposed a dual-ants colony approach to solve FJSP with preventive maintenance.

With an inspiration from the studies on different scheduling problems like single machine scheduling problems, open shop scheduling problems, flow shop scheduling problems, and job-shop scheduling problems under the machine breakdown conditions, research has been focused to solve complex FJSP under machine breakdown. Chaari et al. [28] have proposed a classification of the FJSP under machine breakdown problem into three categories namely proactive methods, reactive methods, and hybrid methods. From Chiang and Fox [40], it is clear that a new external source is required in case of a machine disruption; otherwise, there is high possibility of extension of the schedule completion time. Jensen [61] has proposed a robustness measure and used genetic algorithm to find a robust and flexible schedule. Al-Hinai and ElMekkawy [37] have proposed a two-stage hybrid genetic algorithm (2S-HGA) to solve FJSP under machine breakdown to generate robust and stable schedules. Xiong et al. [33] have used two surrogate measures to solve FJSP under random machine breakdown, and their performance is measured using a multi-objective evolutionary algorithm (MOEA). He et al. [29] have proposed a novel clone immune algorithm (NCIA) to solve FJSP under machine breakdown. Singh et al. [62] have proposed a quantum particle swarm optimization (QPSO) to solve FJSP under machine breakdown using a bi-objective function of makespan and robustness. Nouiri et al. [63] have proposed a two-stage particle swarm optimization (2S-PSO) to generate robust and stable schedules for predictive FJSP under machine breakdown uncertainty.

The present work proposes a 2S-TLBO to solve FJSP under machine breakdown. TLBO is proposed by Rao et al. [64]. TLBO has been applied to different kinds of optimization problems in the past and is known to be one of the efficient algorithms to produce quality solutions in a short computation time [65]. TLBO is applied to different scheduling problems like permutation flow-shop scheduling problem (PFSP) by Xie et al. [66], JSP by Keesari and Rao [67], JSP and FSP by Baykasoglu et al. [68], FJSP with fuzzy processing times by Xu et al. [65], re-entrant flexible FSP by Shen et al. [69], and FJSP by Buddala and Mahapatra [2] and Buddala and Mahapatra [70]. It is found that TLBO is one of the efficient meta-heuristic techniques that can be applied to different scheduling problems. Al-Hinai and ElMekkawy [37] have proposed a two-stage hybrid genetic algorithm (2S-HGA), and Nouiri et al. [63] have proposed a two-stage particle swarm optimization (2S-PSO) to generate robust and stable schedules for predictive FJSP under machine breakdown uncertainty. With an inspiration from the above works, a 2S-TLBO is proposed to solve FJSP under machine breakdown and study its scheduling performance.

3 Proposed two-stage teaching-learning-based optimization approach

3.1 Teaching-learning-based optimization

TLBO, proposed by Rao et al. [64], derives its inspiration from the general teaching-learning process. Student and teacher are the two essential components of a class. A teacher teaches in a class and tries to increase the knowledge of students. Students try to learn from their teacher based on their understanding capability. The students may also discuss with his/her co-student and learn. This is the general process of learning. This is explained in TLBO algorithm as the two modes of learning via teacher (teacher phase) and interaction among the students (student phase). The entire students of the class constitute the population of TLBO. In any iteration, the best student of the class becomes the teacher. Execution of the TLBO is explained in two phases such as teacher phase and student phase.

3.1.1 Teacher phase

A teacher strives his/her best to enhance the knowledge of his/her students to his/her level. But practically, it is impossible as gaining knowledge by a student depends on his/her capacity to learn. In this process, knowledge of each student improves and in turn the average knowledge of the class increases. At any iteration, let Y_{mean} denote the mean knowledge of the class and teacher of the class is denoted as $Y_{teacher}$. Then, increased knowledge of a student is given by the Eq. 1

$$Y_{new\ i} = Y_{old\ i} + r \times \left(Y_{teacher} - \left(T_f \times Y_{mean}\right)\right) \quad (1)$$

where i represents student number, r is a random number between the range [0 1], and T_f is called teaching factor whose value is chosen randomly as one or two. There is no tuning of this teaching factor even though T_f is an algorithm specific parameter of TLBO. This is the main advantage of TLBO. There are no tuning parameters in TLBO. $Y_{new\ i}$ is the new knowledge of the student i after gaining knowledge from the teacher, and $Y_{old\ i}$ is the previous knowledge of the student i. Accept $Y_{new\ i}$ if it gives a better functional value.

3.1.2 Student phase

Students discuss among themselves after a class is taught. In this process, the knowledge of students may increase. At any iteration, let Y_a and Y_b be two students who discuss after the class, $a \neq b$. Then, increased knowledge of the student is given by the Eqs. 2 and 3.

$$Y_{new\ a} = Y_{old\ a} + r \times (Y_a - Y_b) \quad \text{if } F(Y_a) <= F(Y_b) \quad (2)$$

$$Y_{new\ a} = Y_{old\ a} + r \times (Y_b - Y_a) \quad \text{if } F(Y_b) < F(Y_a) \quad (3)$$

where $Y_{new\ a}$ is the new knowledge of the student a after learning from the co-student b. $Y_{old\ a}$ is the previous knowledge of the student a. Accept $Y_{new\ a}$ if it gives a better function value.

Like many meta-heuristics, TLBO also gets trapped at local optimum and loses its diversity after certain number of iterations. To eliminate this drawback and to further enhance the quality of solutions generated by TLBO, a local search technique proposed by Buddala and Mahapatra [71] is used. The proposed local search also maintains diversity in the population. For this purpose, mutation strategy from GA is incorporated to the basic TLBO. For more details on local search and mutation strategy, one can refer Buddala and Mahapatra [71].

3.2 Two-stage teaching-learning-based optimization

In the present problem, FJSP under machine breakdown, the objective is to obtain robust as well as stable schedules. In order to produce the robust and stable schedules, TLBO produces schedules in such a way that makespan of FJSP is minimized in a reasonable number of iterations. This is considered as first stage. Later, TLBO produces schedules in such a way that robust and stable schedules are generated. This is considered as second stage. As the same TLBO is used to minimize two different objectives (one after another) in a single problem, it is termed as two-stage teaching-learning-based optimization (2S-TLBO). A brief description of 2S-TLBO is given as follows.

3.2.1 First stage

There are two stages in the proposed predictive scheduling approach. In the first stage, the task is to minimize the makespan (primary objective function) using TLBO under deterministic conditions with an assumption that there are no uncertainties like machine breakdown. The first stage continues up to prescribed number iterations (say 100 iterations). A switch criterion is introduced in this stage. If the number of iterations reaches the prescribed number of iterations, the criterion helps the algorithm switches into second stage. TLBO is a combination of teacher phase and student phase. Therefore, the combination of teacher phase and student phase constitutes one iteration in both first stage and second stages. This is shown in Fig. 1. The final population generated at the end of first stage is taken as input for the second stage.

3.2.2 Second stage

In the second stage, the objective function is shifted to the bi-objective function (given in Eq. 9). In this stage, machine breakdown condition is taken into account. Here, TLBO minimizes the bi-objective function to generate the required robust and stable schedules. The second stage continues up to prescribed number of iterations (for example 1000 iterations). When the prescribed number of iterations is reached, the termination or stop criterion is met in stage 2. The generalized flowchart of the proposed two-stage TLBO is shown in Fig. 1.

In order to solve the FJSP under machine breakdown condition, first of all, machine breakdown data is to be incorporated into the basic FJSP problem. For this purpose, a non-idle time insertion technique is used. When a machine breakdown occurs in FJSP, rescheduling or re-optimizing the schedule is necessary. For this purpose, a rescheduling technique called modified affected operations research (mAOR) proposed by Subramaniam and Raheja [25] is used. A brief description of (i) FJSP with machine breakdown and how robust and stable schedules are generated using the bi-objective function, (ii) non idle time insertion, and (iii) rescheduling procedure respectively is given in the next section.

4 Flexible job-shop scheduling with machine breakdown

FJSP is an extension of classical JSP. Similar to JSP, FJSP also assigns each operation to an available machine. The additional challenge in FJSP compared to JSP is the sequencing the assigned operations on the machines. A deterministic FJSP is defined as follows.

There are n ($i = 1, 2, 3, \ldots, n$) independent jobs which are to be processed on M ($k = 1, 2, 3, \ldots M$) independent machines. All jobs are available to process at time zero. Each job i has O_i sequence of operations ($j = 1, 2, 3, \ldots, O_i$). For any operation O_{ij}, there exists a machine M_{ijk} such that $M_{ijk} \varepsilon M$. The processing time p_{ijk} of any operation O_{ij} on a given machine k is predefined. Setup times are negligible. In case of setup times if any, they are already included in the processing times. Any operation that is once started cannot be interrupted in between (non-preemption condition). A machine can process not more than one operation at a time (resource constraint). Jobs are different from each other. Machines are different from each other. Jobs transportation time between the operations is negligible. The objective is to find a schedule with minimum makespan value.

Machine breakdown is one of the important uncertainties in production scheduling. This causes a machine unavailable for some duration. Considering all the possible machine breakdowns simultaneously to generate predictive schedule is very difficult task and may consume enormous amount of time due to extensive simulations. Therefore, Nouiri et al. [63] and Al-Hinai and ElMekkawy [37] propose to aggregate all

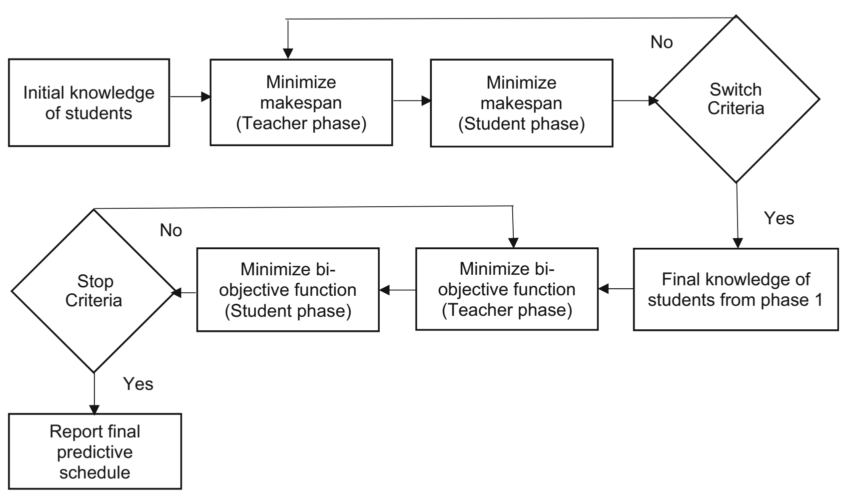

Fig. 1 Flowchart of proposed two-stage TLBO

machine breakdowns into a single breakdown and stability of the schedule can be evaluated accordingly. However, it is an assumed that the data of machine breakdown uncertainty is known in advance. Simulation of a machine breakdown comprises of selection of affected machine, how long that machine is unavailable, and at what time that machine would fail. In this work, the procedure adopted by Al-Hinai and ElMekkawy [37] and Nouiri et al. [63] is used to simulate the machine breakdown generation for FJSP.

The workload of a machine is nothing but maximum duration that a machine is busy. The probability that a machine may fail is directly proportional to the machine workload. Hence, a machine with maximum busy duration is most likely prone to breakdown. The empirical relation for a machine M_k to undergo a breakdown is given in Eq. 4.

$$\rho_k = BTM_k / BTM_{\text{tot}} \tag{4}$$

where BTM_k is the busy duration of machine k and BTM_{tot} is the busy duration of total machine in the shop floor; ρ_k is the probability of a machine M_k to undergo a breakdown ($k = 1, 2, \ldots, M$).

To generate the breakdown time and breakdown duration of a machine, following uniform distributions (eqs. 5 and 6) are used

$$\tau_k = [\alpha_1 BTM_k, \alpha_2 BTM_k] \tag{5}$$

$$\tau_k = [\alpha_1 BTM_k, \alpha_2 BTM_k] \tag{5}$$

$$\tau_{k\ \text{duration}} = [\beta_1 BTM_k, \beta_2 BTM_k] \tag{6}$$

$$\tau_{k\ \text{duration}} = [\beta_1 BTM_k, \beta_2 BTM_k] \tag{6}$$

where τ_k is the breakdown time, $\tau_{k\ \text{duration}}$ is the breakdown duration, BTM_k is the busy duration of machine k, and α and β are the parameters that determine the type of breakdown. For an effective experimentation on FJSP under machine breakdown, we assume two levels of breakdown duration, i.e., low-level breakdown duration and high level breakdown duration as well as two intervals of breakdown time, i.e., early breakdown and late breakdown. If a breakdown occurs during the first half of the machine busy duration, it is called an early breakdown. Otherwise, it is a late breakdown. This leads to four different combinations of breakdown types as presented in the Table 1 [37, 63]. In order to incorporate the machine breakdown data into basic FJSP, a non-idle time insertion technique is used without affecting the sequence of operations. A brief description of non-idle time insertion is given in the Section 4.1.

In general, any optimization problem has an objective function which is to be minimized or maximized. Most of the FJSP research reported till date concentrates either single objective like makespan, tardiness, mean flow time or multi-objective optimization of makespan, total work load, and critical work load combined together. In case of a machine breakdown uncertainty, two more measures are encountered. These are robustness and stability, i.e., a schedule should be generated in such a way that lowest possible makespan is obtained along with a capacity to absorb machine breakdowns such that less number of operations are affected due to breakdown. In the present work, a bi-objective function is used to optimize robustness and stability together.

In the proposed two-stage TLBO, makespan minimization is the important objective during the first stage. Therefore, a schedule is generated in such a way that all the operations take minimum time to complete. Till date, many stability measures have been proposed for FJSP problem. After several computational experiments on different stability measures, Al-Hinai and ElMekkawy [37] have concluded that the following stability measure gives the best results. The stability of an original schedule or a predictive schedule is calculated as the sum of the absolute deviations of operations completion times of the realized schedule and the initial predictive schedule before breakdown. It is given by the Eq. 7 as follows

$$S = \min \sum_{i=1}^{n} \sum_{j=1}^{J_i} \left| PCO_{ij} - RCO_{ij} \right| \tag{7}$$

where S is the stability, n is the number of jobs, J_i is the number of operations of job i, PCO_{ij} is the predicted completion time of jth operation of job i, and RCO_{ij} is the realized completion time of jth operation of job i. P denotes predicted schedule or original schedule before breakdown. R denotes realized schedule or final schedule after breakdown.

Although less work is reported on FJSP under machine breakdown, most of the researchers propose approaches that generate predictive schedules based either on robustness or stability. Very few attempts have been made to optimize both the measures simultaneously. Liu et al. [30] and Liu et al. [31] have tried to optimize robustness and stability measures simultaneously for a single machine problem, with an

Table 1 Different combinations of breakdown

Name	Combination	α_1	α_2	β_1	β_2
BD1	Early, low	0	0.5	0.1	0.15
BD2	Late, low	0.5	1	0.1	0.15
BD3	Early, high	0	0.5	0.35	0.4
BD4	Late, high	0.5	1	0.35	0.4

inspiration from them; Al-Hinai and ElMekkawy [37] have generated optimized schedules for FJSP under machine breakdown by simultaneously considering robustness and stability. A schedule whose makespan deviation is small is called a robust schedule. Such a schedule with minimum operations completion time deviation from the original schedule is called a stable schedule. Using a bi-objective approach, the robustness and stability of a schedule are found using the following bi-objective function (Eq. 8)

$$f = min\ (\gamma \times RC_{max} + (1{-}\gamma) \times S) \tag{8}$$

where $RC_{\max}$ is the realized makespan after machine breakdown, S is the stability measure (Eq. 7), and γ is parameter between the range [0 1]. This is used to assign the relative importance between robustness and stability. The first stage continues up to prescribed number of iterations. Next, the algorithm switches to the second stage. In the second stage, the objective function is shifted to the bi-objective function given in Eq. 8. In Eq. 8, it is observed that $RC_{\max}$ and S are in different scales, i.e., the value of $RC_{\max}$ is comparatively larger than the value of S. Therefore, it is essential to normalize the values of $RC_{\max}$ and S so that predominance effect of either term can be minimized. The normalized bi-objective function of Eq. 8 is given in the Eq. 9 as follows.

$$f = \min\left\{\gamma \times \left(\frac{RC_{\max}{-}LB}{UB{-}LB}\right) + (1{-}\gamma) \times \frac{\sum_{i=1}^{n}\sum_{j=1}^{Ji}\left|PCO_{ij}{-}RCO_{ij}\right|}{RC_{\max}}\right\} \tag{9}$$

where $RC_{\max}$ is the realized makespan after machine breakdown, UB is the upper bound or maximum makespan and LB is the lower bound or minimum makespan, n is the number of jobs, J_i is the number of operations of job i, PCO_{ij} is the predicted completion time of jth operation of job i, and RCO_{ij} is the realized completion time of j_{th} operation of job i. P denotes predicted schedule or original schedule before breakdown. R denotes realized schedule or final schedule after breakdown. The values of UB and LB are taken from Ho and Tay [72] and Palacios et al. [73] respectively. Previous researchers like Nouiri et al. [63] and Al-Hinai and ElMekkawy [37] have not taken the normalization of bi-objective function into account. In the present work, for the better evaluation of the generated schedules, normalized form of bi-objective function is considered (Eq. 9). Also, a sample calculation of f is given in Section 5. Breakdown duration is added to 'UB'.

In order to achieve the objective of generating a robust and stable schedule, a rescheduling approach is necessary to re-optimize the schedule when a machine breakdown occurs. A brief explanation rescheduling strategy is given in the Section 4.2.

4.1 Non-idle time insertion

In idle time insertion, insertion of approximate idle time for each operation in advance is a general strategy for a predictive schedule under possible machine breakdown condition. An initial schedule is generated with an assumption that no machine breakdowns occur. Then, the expected idle time is inserted before the operations. In this process, even though the start and completion times of the operations may vary but the sequence of operations remains the same. The inserted idle times act as buffering times to absorb the adverse effects of machine breakdown. Of course this process increases the makespan of predictive schedule. Generally, two challenges are encountered in insertion of non-idle time. Firstly, it is important to find out the amount of idle times to be inserted and, secondly, to know the correct locations where these idle times are to be inserted.

To overcome the drawback of idle time insertion, a non-idle time insertion method for the FJSP under uncertain machine breakdown is proposed to find out an effective predictive schedule using teaching-learning-based optimization. The main idea is to integrate the available flexible routing of machines with the probability distribution of machine breakdown. Usually, the historical records of the concerned shop floor are used to obtain approximate distribution function for the machine disruptions. Since the historical records of machine breakdown are not available, machine breakdown probabilities are generated using Eq. 4. In order to evaluate the effect of disruptions, robustness and stability measures are evaluated. This process helps us to get solutions that are more robust and stable. This approach is first applied to FJSP under machine breakdown by Al-Hinai and ElMekkawy [37] using hybrid GA (HGA). The same approach is later used by Nouiri et al. [63] using particle swarm optimization (PSO). Application of the non-idle time insertion method produces predictive schedule which is able to assign and sequence the operations on machines in such a way that there is less impact on the overall performance of the schedule.

4.2 Rescheduling procedure

A rescheduling procedure is essential to tackle the machine breakdowns that occur in FJSP environment. When an unexpected machine breakdown occur in a shop floor, the production plan gets affected. Rescheduling is a repair mechanism that is used to repair the production plan. Due to rescheduling, a new executable schedule can be generated in case of any unexpected breakdown events. Especially in the environments like dynamic shop floors, the system should be able to react spontaneously to tackle uncertainties and good quality schedules are to be generated. Hence, a rescheduling procedure is essential to reduce the adverse effects of disruptions. A schedule that is obtained after the implementation of an appropriate rescheduling technique is called a realized schedule. We can find many rescheduling techniques in the literature. Some of them are right shifting rescheduling (RSR), affected operations rescheduling (AOR) by Abumaizar and Svestka [74], a complete regeneration schedule, or modified AOR (mAOR) by Subramaniam and Raheja [25]. For more information about rescheduling, one may go through Subramaniam and Raheja [25] and Dong and Jang [79].

The value of the bi-objective function, realized makespan, and stability measure depends on the rescheduling technique used. Therefore, in order to compare the performance of 2S-TLBO with other algorithms from the literature, we use the same rescheduling technique that is used by Al-Hinai and ElMekkawy [37] for 2S-HGA and Nouiri et al. [63] for 2S-PSO. The rescheduling technique used is mAOR proposed by Subramaniam and Raheja [25]. The main difference between mAOR and AOR is that, mAOR consider not only machine breakdown but also other uncertainties like unexpected arrival of jobs and uncertain processing times. The idea of this method is to maintain the sequence of jobs that are to be processed on each machine same as that of the original schedule which are affected directly or indirectly due to machine breakdown [74]. The reason to maintain the same sequence of operations in mAOR is to avoid setup costs that arise due to changes in sequence. Therefore, operations are affected only in terms of starting and completion times whenever there is a machine breakdown. Thus, after breakdowns, the realized operations completion times are calculated as follows (Eq. 10)

$$C_{ijk} = \begin{cases} S_{ijk} + P_{ijk} & \text{for unaffected operations} \\ S_{ijk} + P_{ijk} + \tau_{k\ \text{duration}} & \text{for afftected operations} \end{cases} \quad (10)$$

where P_{ijk} is the processing time on machine k for ith job's jth operation. C_{ijk} and S_{ijk} are the completion and starting times of ith job's jth operation on machine k. $\tau_{k\ \text{duration}}$ is the duration of machine unavailability due to breakdown.

5 Results and discussion

In order to test the performance and effectiveness of the proposed two-stage TLBO, experiments have been conducted on the standard benchmark problems of Kacem's [75, 76] and Brandimarte's [77, 78] data sets. The problems contain both partial FJSP's and total FJSP's. Out of the total 14 problems, Kacem's 10 × 10, 10 × 7, and 15 × 10 come under total flexibility. Remaining 11 problems that consist of Kacem's 8 × 8 and Brandimarte's MK01 to MK10 come under partial flexibility. In all these problems, the job count varies from 8 to 20, machine count varies from 6 to 15, and operation count varies from 27 to 232. Experiments have been conducted on MATLAB platform on a 4-GB ram i7 processor running at 3.40 GHz on windows 7 platform.

To study the performance of proposed two-stage TLBO in generating the predictive schedules, experiments have been conducted on the standard benchmark problems and the results are compared with the results of previous algorithms 2S-HGA (proposed by [37]) and 2S-PSO (proposed by [63]). In this process, each benchmark problem per instance is subjected to five replications of four different breakdowns as illustrated in Table 1. Next, each predictive schedule generated is subjected to 400 random machine breakdowns. This results in 5 × 4 × 400 = 8000 test problems per instance.

In the proposed TLBO algorithm, the parameter values are chosen as follows: population size = 100 and maximum iterations = 100. Soon after 100 generations in the first stage, the algorithm switches to the second stage to optimize the bi-objective function. The value of parameter γ can be any value in the range [0 1]. The value of γ depends on the decision maker's view point. γ acts as trade-off value between robustness and stability. If the decision maker assigns a value $\gamma = 1$, it implies that decision maker wants to minimize only makespan with stability having zero importance. If the decision maker assigns a value $\gamma = 0$, it implies that decision maker wants a stable schedule which means that there are no deviations between the realized and predicted schedules operations' starting and completion times. As discussed earlier, in order to compare with previous algorithms, we use the same value of $\gamma = 0.6$ for TLBO, which is used by previous researchers (for HGA and PSO).

In order to evaluate the performance of any algorithm that generates a predictive schedule, using robustness and stability measures, Al-Hinai and ElMekkawy [37] proposed two important measures called average realized makespan improvement percentage (denoted by AMSRI) and average stability improvement percentage (denoted by ASTBI) respectively. The equations to find AMSRI and ASTBI are given in Eqs. 11 and 12 respectively.

$$\text{AMSRI} = \frac{\sum_{q=1}^{5}\sum_{p=1}^{400} RMSR_{(q)p} - \sum_{q=1}^{5}\sum_{p=1}^{400} DMSR_{(q)p}}{\sum_{q=1}^{5}\sum_{p=1}^{400} DMSR_{(q)p}} \times 100 \quad (11)$$

$$\text{ASTBI} = \frac{\sum_{q=1}^{5}\sum_{p=1}^{400} RSTB_{(q)p} - \sum_{q=1}^{5}\sum_{p=1}^{400} DSTB_{(q)p}}{\sum_{q=1}^{5}\sum_{p=1}^{400} DSTB_{(q)p}} \times 100 \quad (12)$$

where q is the replication number of predictive schedule and p is the breakdown number. *RMSR* is the realized makespan after a breakdown using robust method for the obtained schedule. *DMSR* is the realized makespan after breakdown using deterministic method for the obtained schedule. *RSTB* is the stability of the schedule obtained with robust method. *DSTB* is the stability of the schedule obtained with deterministic method. Here, the deterministic method is nothing but the TLBO algorithm which minimizes makespan in the first stage.

In Table 4, the average of obtained realized makespan improvement percentage (AMSRI) results of TLBO are compared with the results of PSO by Nouiri et al. [63] and HGA by Al-Hinai and ElMekkawy [37]. In Table 5, the average stability improvement percentage (ASTBI) results of TLBO are compared with the results of PSO and HGA. In Tables 4 and 5, there are 14 columns. First column denotes the name of the instance. Second column gives the size of the problem. Columns three to five gives the results comparison for the breakdown type 1 (BD1). Columns six to eight gives the results comparison for the breakdown type 2 (BD2). Columns nine to 11 give the results comparison for the breakdown type 3 (BD3). Columns 12 to 14 give the results comparison for the breakdown type 4 (BD4). And numbers in italics indicate the best values. The negative values in the tables indicate that these algorithms are able to generate improvised results. On the other hand, positive values indicate that degraded results are generated.

A close look at Table 4 clearly shows that TLBO outperforms PSO and HGA. In BD1, TLBO gives best results to four problems (MK02, MK05, MK06, and MK10) compared to PSO and HGA. In terms of degraded results, all the three algorithms give degraded results in two problems each: TLBO in MK03 and MK07, PSO in MK01 and MK10. and HGA in MK04 and MK10. In terms of average AMSRI value, TLBO (− 3.7411) outperforms PSO (− 3.38) and HGA (− 2.31) in BD1. In BD2, TLBO gives best results to four problems (MK01, MK05, MK08, and MK10). In terms of degraded results, TLBO gives degraded results for six problems (KCM3, MK03, MK04, MK06, MK07, and MK08) which is less in number than seven problems of PSO (MK01, MK03, MK04, MK05, MK07, MK08, and MK10). But HGA also generates equal number of degraded results to that of TLBO (KCM3, MK01, MK04, MK06, MK08, and MK10). In terms of average AMSRI value, TLBO (− 1.4573) outperforms PSO (− 1.41) and HGA (− 1.22) in BD2. In BD3, TLBO gives best results to six problems (KCM2, KCM3, MK01, MK02, MK05, and MK06). In terms of degraded results, TLBO gives degraded results to two problems (MK04 and MK07) which are much less than PSO (KCM3, MK01, MK02, MK03, and MK04). But HGA generates zero degraded results. Hence, HGA outperforms TLBO and PSO in terms of degraded solutions in BD3. But, in terms of average AMSRI value, TLBO (− 7.0245) outperforms PSO (− 2.11) and HGA (− 6.48) in BD3. In BD4, TLBO gives best results only to two problems. Both in terms of degraded results and in terms of AMSRI values, PSO (− 7.45) shows a better performance than TLBO (− 5.4195) and HGA (− 6.71). Therefore, in BD4, PSO outperformed the other two algorithms. Finally, as TLBO gives best AMSRI values in three (BD1, BD2, and BD3) out of four breakdowns, we can conclude that TLBO is superior to PSO and HGA.

A close look at Table 5 clearly shows that TLBO outperforms PSO and HGA. In BD1, TLBO gives best stability values to ten problems. The average stability value of TLBO (− 84.5534) clearly outperforms PSO (− 51.96) and HGA (− 78.84). In BD2, TLBO gives best stability values to 12 problems. The average stability value of TLBO (− 79.6198) clearly outperforms PSO (− 27.50) and HGA (− 45.54). In BD3, TLBO gives best stability values to 11 problems. The average stability value of TLBO (− 92.345) clearly outperforms PSO (− 21.95) and HGA (− 62.48). In BD4, TLBO gives best stability values to 11 problems. The average stability value of TLBO (− 88.3676) clearly outperforms PSO (− 21.96) and HGA (− 44.14). In all the four breakdown types, TLBO outperforms PSO and HGA. Thus, we can conclude that TLBO is superior to PSO and HGA in terms of ASTBI values.

Based on the computational results from Table 4 (average AMSRI values) and Table 5 (average ASTBI values), Figs. 2 and 3 show the graphical representation of efficiency of proposed TLBO versus PSO and HGA. In Figs. 2 and 3, *X*-axis denotes the BD type and *Y*-axis denotes AMSRI and ASTBI values respectively. From graphs, it is clear that in the algorithm whose performance is superior, algorithm's curve will be more close to the *X*-axis. Even though Nouiri et al. [63] claims that PSO is superior to HGA in terms of AMSRI values (Fig. 2), the performance of PSO in terms of ASTBI is poor. This is clearly shown in Fig. 3. The curve of PSO (red colour) is above the HGA (blue colour). The curve of TLBO (black colour) which is below the other two curves (PSO and HGA) and much closer to *X*-axis in both the figures shows the superiority of TLBO.

To confirm whether the obtained results of TLBO are statistically significant or not, we further examine the results of TLBO with analysis of variance (one-way ANOVA) test. Tables 6 and 7 show the *P* values and *F* ratios of the obtained one-way ANOVA results, which gives the details of effects and interactions between the factors of robustness and stability measures, breakdown type, and considered test cases on the predictive schedules relative

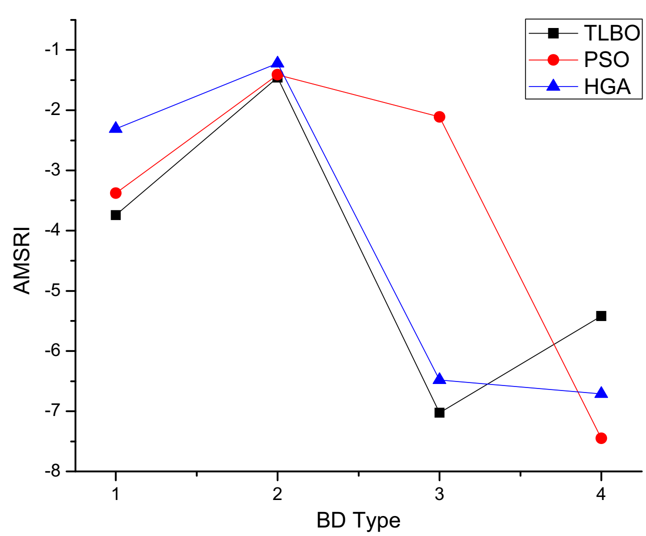

Fig. 2 Algorithm performance for each BD type on AMSRI

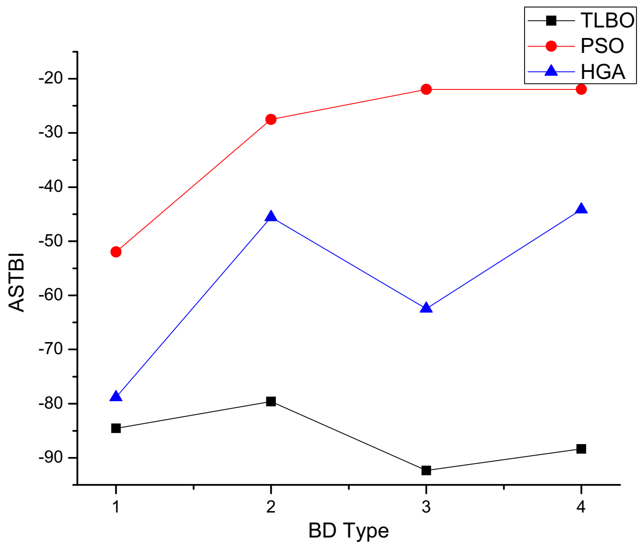

Fig. 3 Algorithm performance for each BD type on ASTBI

Table 3 AMSRI and ASTBI values of case study problem

BD type	AMSRI	ASTBI
BD1	− 3.87363	− 82.4706
BD2	− 2.18218	− 44.1253
BD3	− 8.87935	− 96.2372
BD4	− 9.56933	− 65.3791

quality. If the P value is less than 0.05, the effects are considered significant in our study. Therefore, values less than 0.05 are best and are indicated in italic numbers.

Tables 6 and 7 convey that there is a significant effect on both P value and F ratio by the BD type because BD1, BD2, BD3, and BD4 have P value less than 0.05. Therefore, the 2S-TLBO is clearly different from PSO and HGA statistically. Hence, compared to other approaches, the 2S-TLBO gives a better performance.

Table 2 Processing times of the mechanical workshop

Jobs	Operations	Machine 1	Machine 2	Machine 3	Machine 4	Machine 5	Machine 6
1	O11	12	18		18		16
	O12		8	6	20	20	
	O13		8			18	10
	O14	14		16	16		
	O15	8	8	10	10	6	12
	O16			20		16	18
2	O21	18		8	20	14	
	O22		18	12	9		15
	O23	7		20	12	18	
3	O31	18	20	8	7	6	
	O32		7	8		20	
	O33	7	10		14		20
	O34		8	20	16	15	
	O35	20		18	10	10	15
4	O41	22		19	16	20	
	O42		13	10	18	10	18
	O43	6	10				20
	O44	12		8		8	
	O45	20	12	20	15		
5	O51		8		10	15	10
	O52	12		15	20		8
	O53	16	15			20	10
	O54		6	8	8	15	
	O55	15			10		12
	O56	13	5	13	15	20	7
6	O61		14			9	10
	O62			12	15		9
	O63	13	7		16	9	
	O64	20		13			17

Empty box indicate that the machine cannot process that operation

Table 4 AMSRI computational results for all types of breakdown

AMSRI

Instance	Size	BD1			BD2			BD3			BD4		
		TLBO	PSO	HGA	TLBO	PSO	HGA	TLBO	PSO	HGA	TLBO	PSO	HGA
KCM1	8 × 8	− 3.5691	*− 7.98*	− 1.35	− 0.3335	*− 6.95*	− 4.37	− 7.4787	*− 19.86*	− 13.33	− 8.772	*− 27.58*	2.47
KCM2	10 × 10	− 5.618	*− 6.97*	− 5.41	− 5.618	*− 6.44*	− 5.41	*− 15.062*	− 3.72	− 6.98	− 6.8329	*− 13.54*	− 9.52
KCM3	15 × 10	0	*− 7.98*	− 1.64	0.0714	*− 10.41*	3.28	*− 5.8157*	10.61	− 2.86	− 10.224	*− 11.39*	− 1.76
KCM4	10 × 7	− 3.7852	NA	NA	− 10.238	NA	NA	− 13.341	NA	NA	− 19.191	NA	NA
MK01	10 × 6	− 3.1871	0.77	*− 4.7*	*− 2.2499*	4.8	0.91	*− 10.497*	0.93	− 9.39	*− 9.4793*	− 6.27	− 8.8
MK02	10 × 6	*− 8.3636*	− 1.64	0	− 8.4715	*− 9.79*	0	*− 10.65*	1.58	0	− 7.5061	− 0.83	*− 7.73*
MK03	15 × 8	4.8092	− 2.24	*− 4.45*	4.2458	6.92	*− 7.27*	− 1.5543	6.16	*− 5.24*	− 1.4187	*− 14.96*	− 13.18
MK04	15 × 8	− 3.053	*− 6.46*	0.29	6.07655	3.72	*0.3*	5.2718	12.73	*− 8.74*	2.0666	*− 14.49*	− 13.83
MK05	15 × 4	*− 13.481*	− 3.56	− 1.33	*− 7.7019*	0.54	− 1.05	*− 17.002*	− 9.35	− 7.32	− 9.8583	− 6.54	*− 13.87*
MK06	10 × 15	*− 11.979*	− 2.45	− 1.31	2.43705	*− 2.65*	0.25	*− 15.56*	− 8.21	− 7.28	*− 7.9347*	− 6.18	− 4.23
MK07	20 × 5	5.54577	− 0.43	*− 1.87*	3.71215	1.32	*− 2.5*	5.24253	*− 2.76*	− 0.31	2.70705	− 5.76	*− 6.28*
MK08	20 × 10	− 3.9603	− 2.31	*− 4.05*	*1.50863*	2.32	1.72	− 0.9214	− 4.2	*− 8.4*	2.30295	− 0.54	*− 1.93*
MK09	20 × 10	− 4.0348	− 3.38	*− 4.23*	− 3.3256	*− 3.42*	− 2.51	− 5.2921	*− 6.2*	− 5.4	− 1.3088	− 1.98	*− 5.58*
MK10	20 × 15	*− 1.6992*	1.23	2.87	*− 0.5152*	1.65	0.74	− 6.1359	− 5.23	*− 9*	− 0.423	− 1.76	*− 3.11*
Average		*− 3.7411*	− 3.38	− 2.31	*− 1.4573*	− 1.41	− 1.22	*− 7.0245*	− 2.11	− 6.48	− 5.4195	*− 7.45*	− 6.71

Numbers in italics indicate the best values

NA not applicable

Table 5 ASTBI computational results for all types of breakdown

ASTBI

Instance	Size	BD1			BD2			BD3			BD4		
		TLBO	PSO	HGA	TLBO	PSO	HGA	TLBO	PSO	HGA	TLBO	PSO	HGA
KCM1	8 × 8	*− 91.8025*	− 39.03	− 73.33	*− 46.2396*	− 37.105	− 37.12	*− 99.8775*	− 8.74	− 83.96	*− 99.7937*	− 24.82	− 86.96
KCM2	10 × 10	*− 100*	− 90.45	*− 100*	*− 100*	− 82.5	− 90.91	*− 100*	− 17.22	− 46.15	*− 95.5*	− 8.7	− 60
KCM3	15 × 10	− 42.8572	− 98.16	*− 96.97*	− 1.9704	− 13.46	*− 58.26*	*− 99.5217*	12.62	− 82.73	*− 95.5438*	− 42.94	− 37.34
KCM4	10 × 7	− 91.0313	NA	NA	− 100	NA	NA	− 100	NA	NA	− 98.4532	NA	NA
MK01	10 × 6	*− 93.4033*	− 13.26	− 62.64	*− 90.6*	− 16.65	− 81.74	*− 96.501*	8.92	− 69.71	*− 92.8107*	− 19.26	− 44.67
MK02	10 × 6	*− 94.2074*	− 17.23	− 55.61	*− 80.5091*	− 16.21	− 5.89	*− 98.3346*	− 21.74	− 20.94	*− 88.6361*	− 34.63	− 11.3
MK03	15 × 8	− 57.0089	− 26.95	*− 85.85*	*− 98.5251*	8.49	− 66.79	*− 97.9715*	− 33.96	− 70.51	*− 97.2607*	− 50.83	− 49.82
MK04	15 × 8	− 32.5381	− 15.58	*− 49.09*	*− 89.6951*	10.85	− 32.63	*− 72.922*	− 10.4	*− 96.38*	− 73.4595	*− 99.15*	− 92.11
MK05	15 × 4	*− 96.4029*	− 65.76	− 61.02	*− 77.8158*	− 4.23	− 9.01	*− 77.4773*	*− 89.34*	− 70.19	− 67.1832	*− 78.26*	− 60.6
MK06	10 × 15	*− 99.0561*	− 60.65	− 58.78	*− 98.0898*	− 78.34	− 42.61	*− 76.7447*	− 42.56	− 66.23	*− 91.2268*	− 43.28	− 36.99
MK07	20 × 5	*− 96.5511*	− 64.21	− 83.42	*− 74.141*	− 23.35	− 40.26	*− 89.2773*	− 26.23	− 31.18	*− 79.8806*	− 22.87	− 32.28
MK08	20 × 10	*− 98.2086*	− 90.23	− 86.57	*− 64.2346*	− 50.45	− 31.54	*− 94.6847*	− 43.23	− 61.78	*− 86.5854*	− 43.65	− 33.38
MK09	20 × 10	*− 93.7733*	− 70.32	− 66.6	*− 96.072*	− 23.65	− 46.96	*− 98.1707*	− 50.23	− 36.99	*− 88.1663*	− 11.65	− 26.34
MK10	20 × 15	*− 96.9065*	− 50.21	− 58.23	*− 96.7853*	− 30.87	− 48.32	*− 98.88*	− 49.76	− 75.88	*− 82.6462*	− 3.76	− 2.04
AVERAGE		*− 84.5534*	− 51.96	− 78.84	*− 79.6198*	− 27.5	− 45.54	*− 92.345*	− 21.95	− 62.48	*− 88.3676*	− 21.96	− 44.14

Numbers in italics indicate the best values

NA not applicable

Table 6 Comparison of one-way ANOVA (TLBO with PSO)

BD type	AMSRI		ASTBI	
	P value	*F* ratio	*P* value	*F* ratio
BD1	*0.000228*	3.816948	*0.003451*	2.844463
BD2	*1.76E−09*	8.974538	*0.002106*	3.018287
BD3	*8.39E−07*	6.037565	0.983167	0.334229
BD4	*0.000254*	3.777871	0.974955	0.366282

Table 7 Comparison of one-way ANOVA (TLBO with HGA)

BD type	AMSRI		ASTBI	
	P value	*F* ratio	*P* value	*F* ratio
BD1	*0.000663*	3.429669	*0.001168*	3.227152
BD2	*1.59E−07*	6.770441	*6.04E−05*	4.312768
BD3	*1.8E−06*	5.713247	0.843895	0.599601
BD4	*0.000494*	3.535345	0.757785	0.697017

5.1 Case study

In order to demonstrate the practical application of the present work in a real life flexible job-shop scheduling problem, a case study of a mechanical workshop has been carried out using the data from Jiang et al. [79]. The shop floor consists of six machines on which six jobs have to be processed with a total number of 29 operations. Minimum operations of a job are three while the maximum operations of a job are six. The processing times of these operations are given in the Table 2, and the results are given in Table 3.

5.2 Sample calculation to evaluate *f*

In order to demonstrate the calculation of *f*, readings for a single machine breakdown of case study problem under BD1 condition for a random population are provided. Deterministic makespan at the end of first stage is 64. The second stage was run for 100 iterations after introducing uncertainties in machine breakdown. During the optimization of bi-objective function in the second stage, the value of realized makespan ($RC_{\max}$) is 67 at 100th iteration and the value of stability measure $|PCO_{ij}-RCO_{ij}|$ is 4. Lower bound (LB) on makespan = 58, and upper bound (UB) for makespan = 124. Substituting the values in Eq. 10, we get

$$f=\left\{\gamma\times\left(\frac{RC_{\max}-LB}{UB-LB}\right)+(1-\gamma)\times\frac{\sum_{i=1}^{n}\sum_{j=1}^{Ji}\left|PCO_{ij}-RCO_{ij}\right|}{RC_{\max}}\right\}$$

$$f=\left\{0.6\times\left(\frac{67-58}{124-58}\right)+(1-0.6)\times\frac{4}{67}\right\}$$

$$f=[0.081+0.024]$$

$$f=0.105$$

AMSRI and ASTBI are calculated over 100 iterations and the results are shown in Table 3. From Table 3, it is found that all the AMSRI and ASTBI values are negative. It indicates that the proposed two-stage teaching-learning-based optimization algorithm gives improvised realized makespan and stability to FJSP of the above mechanical workshop under machine breakdown condition.

6 Conclusions

In this work, FJSP under machine breakdown uncertainty is chosen for the study. For this work, we propose a two-stage teaching-learning-based optimization to tackle this problem. The main objective is to generate a predictive schedule with minimum makespan which is simultaneously both robust and stable when an unexpected machine failure disruption occurs. The robust and stable schedule acts as an alternative schedule to an existing schedule that reduces the adverse effect of machine breakdown. Therefore, in order to generate robust and stable schedule, an initial schedule is required. The initial schedule is generated in the first stage. Using this initial schedule from first stage, a robust and stable schedule is generated in the second stage. Thus, the problem is solved in two stages in the proposed solution methodology. Computational experiments have been conducted on Kacem's and Brandimarte's data instances. Analysis of obtained 2S-TLBO results is carried out using one-way ANOVA to test the significance of obtained TLBO results. The results of computational experiments clearly show that proposed 2S-TLBO generates superior predictive schedules when compared to 2S-PSO proposed by Nouiri et al. [63] and 2S-HGA proposed by Al-Hinai and ElMekkawy [37]. The one-way ANOVA test shows that the obtained TLBO results are statistically significant from other algorithms.

In future, attempt may be made on proposing a hybrid approaches (i.e., predictive-reactive technique) based TLBO to solve FJSP under machine breakdown. The work can also be extended to solve the problem in one stage rather than using a two-stage approach. The work can also be extended to generate robust and stable schedules by aiming not only machine breakdown but multiple uncertainties together as a multi-objective optimization problem.

Acknowledgements The authors express hearty thanks to the editor(s) and anonymous reviewers of IJAMT for their careful reading and suggestions in improving the quality of the paper.

References

1. Bagheri A, Zandieh M, Mahdavi I, Yazdani M (2010) An artificial immune algorithm for the flexible job-shop scheduling problem. Future Gener Comput Syst 26:533–541
2. Buddala R, Mahapatra SS (2016) An effective teaching learning based optimization for flexible job shop scheduling. In Electr Electron Optim Tech (ICEEOT), Int Conf 3087–3092 IEEE
3. Gao J, Sun L, Gen M (2008) A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. Comput Oper Res 35:2892–2907
4. Gao KZ, Suganthan PN, Pan QK, Chua TJ, Cai TX, Chong CS (2016) Discrete harmony search algorithm for flexible job shop scheduling problem with multiple objectives. J Intell Manuf 27: 363–374
5. Li X, Gao L (2016) An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem. Int J Prod Econ 174:93–110
6. Liouane N, Saad I, Hammad S, Borne P (2007) Ant systems & local search optimization for flexible job shop scheduling production. Int J Comput Commun Control 2:174–184
7. Rahmati SHA, Zandieh M (2012) A new biogeography-based optimization (BBO) algorithm for the flexible job shop scheduling problem. Int J Adv Manuf Tech 58:1115–1129
8. Singh MR, Mahapatra SS (2016) A quantum behaved particle swarm optimization for flexible job shop scheduling. Comput Ind Eng 93:36–44
9. Wang L, Zhou G, Xu Y, Wang S, Liu M (2012) An effective artificial bee colony algorithm for the flexible job-shop scheduling problem. Int J Adv Manuf Tech 60:303–315
10. Wu X, Wu S (2017) An elitist quantum-inspired evolutionary algorithm for the flexible job-shop scheduling problem. J Intell Manuf 28:1441–1457
11. Xing LN, Chen YW, Wang P Zhao QS, Xiong J (2010) A knowledge-based ant colony optimization for flexible job shop scheduling problems. Appl Soft Comput 10:888–896
12. Yuan Y, Xu H, Yang J (2013) A hybrid harmony search algorithm for the flexible job shop scheduling problem. Appl Soft Comput 13: 3259–3272
13. Zhang G, Gao L, Shi Y (2011) An effective genetic algorithm for the flexible job-shop scheduling problem. Expert Syst Appl 38: 3563–3573
14. Pinedo, M. (2008) Scheduling: theory, algorithms and systems
15. Deng Q, Gong G, Gong X, Zhang L, Liu W, Ren Q (2017) A bee evolutionary guiding nondominated sorting genetic algorithm II for multiobjective flexible job-shop scheduling. Comput Intell Neurosci 2017:1–20. https://doi.org/10.1155/2017/5232518
16. Karthikeyan S, Asokan P, Nickolas S, Page T (2015) A hybrid discrete firefly algorithm for solving multi-objective flexible job shop scheduling problems. Int J Bio-Inspired Comput 7:386–401
17. Kemmoe-Tchomte S, Lamy D, Tchernev N (2017) An effective multi-start multi-level evolutionary local search for the flexible job-shop problem. Eng Appl Artif Intell 62:80–95
18. Li JQ, Pan QK, Liang YC (2010) An effective hybrid tabu search algorithm for multi-objective flexible job-shop scheduling problems. Comput Ind Eng 59:647–662
19. Li JQ, Pan QK, Tasgetiren MF (2014) A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities. Appl Math Model 38: 1111–1132
20. Perez MA, Raupp FM (2016) A Newton-based heuristic algorithm for multi-objective flexible job-shop scheduling problem. J Intell Manuf 27:409–416
21. Singh MR, Singh M, Mahapatra SS, Jagadev N (2016) Particle swarm optimization algorithm embedded with maximum deviation theory for solving multi-objective flexible job shop scheduling problem. Int J Adv Manuf Tech 85:2353–2366
22. Xia W, Wu Z (2005) An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. Comput Ind Eng 48:409–425
23. Xing LN, Chen YW, Yang KW (2009a) Multi-objective flexible job shop schedule: design and evaluation by simulation modeling. Appl Soft Comput 9:362–376
24. Xing LN, Chen YW, Yang KW (2009b) An efficient search method for multi-objective flexible job shop scheduling problems. J Intell Manuf 20:283–293
25. Subramaniam V, Raheja AS (2003) mAOR: a heuristic-based reactive repair mechanism for job shop schedules. Int J Adv Manuf Techn 22:669–680
26. Ourari S, Berrandjia L, Boulakhras R, Boukciat A, Hentous H (2015) Robust approach for centralized job shop scheduling: sequential flexibility. IFAC-PapersOnLine 48:1960–1965
27. Vieira GE, Herrmann JW, Lin E (2003) Rescheduling manufacturing systems: a framework of strategies, policies, and methods. J Scheduling 6:39–62
28. Chaari T, Chaabane S, Aissani N, Trentesaux D (2014) Scheduling under uncertainty: survey and research directions. In Adv Logist Transport (ICALT), 2014 Int Conf 229–234 IEEE
29. He W, Sun D, Liao X (2013) Applying novel clone immune algorithm to solve flexible job shop problem with machine breakdown. J Inf Comput Sci 10:2783–2797
30. Liu N, Abdelrahman MA, Ramaswamy S (2007a) A complete multiagent framework for robust and adaptable dynamic job shop scheduling. IEEE Trans Syst Man Cybern Part C Appl Rev 37:904–916
31. Liu H, Abraham A, Grosan C (2007b) A novel variable neighborhood particle swarm optimization for multi-objective flexible job-shop scheduling problems. In Digital Inf Manage 2007 ICDIM'07. 2nd Int Conf 1:138–145 **IEEE**
32. Bidot J, Vidal T, Laborie P, Beck JC (2009) A theoretic and practical framework for scheduling in a stochastic environment. J Scheduling 12(3):315–344
33. Xiong J, Xing LN, Chen YW (2013) Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. Int J Prod Econ 141:112–126
34. Wu SD, Storer RH, Pei-Chann C (1993) One-machine rescheduling heuristics with efficiency and stability as criteria. Comput Oper Res 20:1–14
35. Goren S, Sabuncuoglu I (2009) Optimization of schedule robustness and stability under random machine breakdowns and processing time variability. IIE Trans 42:203–220
36. Jensen MT (2001) Improving robustness and flexibility of tardiness and total flow-time job shops using robustness measures. Appl Soft Comput 1:35–52
37. Al-Hinai N, ElMekkawy TY (2011) Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm. Int J Prod Econ 132:279–291
38. Wang K, Choi SH (2012) A decomposition-based approach to flexible flow shop scheduling under machine breakdown. Int J Prod Res 50:215–234
39. Al-Hinai N, ElMekkawy TY (2012) Solving the flexible job shop scheduling problem with uniform processing time uncertainty. World Acad Sci, Eng Tech 64:996–1001
40. Chiang, W Y, Fox MS (1990) Protection against uncertainty in a deterministic schedule. In Fourth Int Conf Expert Syst Prod Oper Manage South California, USA 17:
41. Dalfard VM, Mohammadi G (2012) Two meta-heuristic algorithms for solving multi-objective flexible job-shop scheduling with parallel machine and maintenance constraints. Comput Math Appl 64: 2111–2117

42. Mouelhi-Chibani W, Pierreval H (2010) Training a neural network to select dispatching rules in real time. Comput Ind Eng 58:249–256
43. Zbib N, Pach C, Sallez Y, Trentesaux D (2012) Heterarchical production control in manufacturing systems using the potential fields concept. J Intell Manuf 23:1649–1670
44. Cowling PI, Ouelhadj D, Petrovic S (2004) Dynamic scheduling of steel casting and milling using multi-agents. Prod Plann Control 15:178–188
45. Gao KZ, Suganthan PN, Tasgetiren MF, Pan QK, Sun QQ (2015) Effective ensembles of heuristics for scheduling flexible job shop problem with new job insertion. Comput Ind Eng 90:107–117
46. Wu LH, Chen X, Chen XD, Chen QX (2009) The research on proactive-reactive scheduling framework based on real-time manufacturing information. In Mater Sci Forum 626:789–794 **Trans Tech Publ**
47. Li W, Cao J (1995) Stochastic scheduling on a single machine subject to multiple breakdowns according to different probabilities. Oper Res Lett 18:81–91
48. Kasap N, Aytug H, Paul A (2006) Minimizing makespan on a single machine subject to random breakdowns. Oper Res Lett 34:29–36
49. Allahverdi A (1995) Two-stage production scheduling with separated set-up times and stochastic breakdowns. J Oper Res Soc 46:896–904
50. Allahverdi A (1996) Two-machine proportionate flowshop scheduling with breakdowns to minimize maximum lateness. Comp Oper Res 23:909–916
51. Allahverdi A, Mittenthal J (1994) Two-machine ordered flowshop scheduling under random breakdowns. Math Comput Modell 20:9–17
52. Allahverdi A, Mittenthal J (1998) Dual criteria scheduling on a two-machine flowshop subject to random breakdowns. Int Trans Oper Res 5:317–324
53. Alcaide D, Rodriguez-Gonzalez A, Sicilia J (2002) An approach to solve the minimum expected makespan flow-shop problem subject to breakdowns. Eur J Oper Res 140:384–398
54. Alcaide D, Rodriguez-Gonzalez A, Sicilia J (2005) A heuristic approach to minimize expected makespan in open shops subject to stochastic processing times and failures. Int J Flex Manuf Syst 17:201–226
55. Luh PB, Chen D, Thakur LS (1999) An effective approach for job-shop scheduling with uncertain processing requirements. IEEE Trans Rob Autom 15:328–339
56. Lei D (2011) Scheduling stochastic job shop subject to random breakdown to minimize makespan. Int J Adv Manuf Tech 55:1183–1192
57. Ahmadi E, Zandieh M, Farrokh M, Emami SM (2016) A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms. Comput Oper Res 73:56–66
58. Park J, Mei Y, Nguyen S, Chen G, Zhang M (2017). Investigating the generality of genetic programming based hyper-heuristic approach to dynamic job shop scheduling with machine breakdown. In Australasian Conf Artif Life Comput Intell 301–313 Springer
59. Zandieh M, Khatami AR, Rahmati SH (2017) Flexible job shop scheduling under condition-based maintenance: improved version of imperialist competitive algorithm. Appl Soft Comput 58:449–464
60. El Khoukhi F, Boukachour J, Alaoui AE (2017) The "dual-ants colony": a novel hybrid approach for the flexible job shop scheduling problem with preventive maintenance. Comput Ind Eng 106:236–255
61. Jensen MT (2003) Generating robust and flexible job shop schedules using genetic algorithms. IEEE Trans Evol Comput 7:275–288
62. Singh MR, Mahapatra SS, Mishra R (2014) Robust scheduling for flexible job shop problems with random machine breakdowns using a quantum behaved particle swarm optimisation. Int J Serv Oper Manage 20:1–20
63. Nouiri M, Bekrar A, Jemai A, Trentesaux D, Ammari AC, Niar S (2017) Two stage particle swarm optimization to solve the flexible job shop predictive scheduling problem considering possible machine breakdowns. Comput Ind Eng 112:595–606
64. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43:303–315
65. Xu Y, Wang L, Wang SY, Liu M (2015) An effective teaching–learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time. Neurocomputing 148:260–268
66. Xie Z, Zhang C, Shao X, Lin W, Zhu H (2014) An effective hybrid teaching–learning-based optimization algorithm for permutation flow shop scheduling problem. Adv Eng Softw 77:35–47
67. Keesari HS, Rao RV (2014) Optimization of job shop scheduling problems using teaching-learning-based optimization algorithm. Opsearch 51:545–561
68. Baykasoglu A, Hamzadayi A, Köse SY (2014) Testing the performance of teaching–learning based optimization (TLBO) algorithm on combinatorial problems: flow shop and job shop scheduling cases. Inf Sci 276:204–218
69. Shen JN, Wang L, Zheng HY (2016) A modified teaching–learning-based optimisation algorithm for bi-objective re-entrant hybrid flowshop scheduling. Int J Prod Res 54:3622–3639
70. Buddala R, Mahapatra SS (2018) An integrated approach for scheduling flexible job-shop using teaching–learning-based optimization method. J Ind Eng Int 1–12. https://doi.org/10.1007/s40092-018-0280-8
71. Buddala R, Mahapatra SS (2017) Improved teaching–learning-based and JAYA optimization algorithms for solving flexible flow shop scheduling problems. J Ind Eng Int 14:555–570. https://doi.org/10.1007/s40092-017-0244-4
72. Ho NB, Tay JC (2004) GENACE: an efficient cultural algorithm for solving the flexible job-shop problem. In Evolutionary computation, 2004. CEC2004. Congress on 2004 (Vol. 2, pp. 1759–1766). IEEE
73. Palacios JJ, González MA, Vela CR, González-Rodríguez I, Puente J (2015) Genetic tabu search for the fuzzy flexible job shop problem. Comput Oper Res 54:74–89
74. Abumaizar RJ, Svestka JA (1997) Rescheduling job shops under random disruptions. Int J Prod Res 35:2065–2082
75. Kacem I, Hammadi S, Borne P (2002a) Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. IEEE Trans Syst Man Cybern Part C Appl Rev 32:1–13
76. Kacem I, Hammadi S, Borne P (2002b) Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic. Math Comput Simul 60:245–276
77. Dong YH, Jang J (2012) Production rescheduling for machine breakdown at a job shop. Int J Prod Res 50:2681–2691
78. Brandimarte P (1993) Routing and scheduling in a flexible job shop by tabu search. Ann Oper Res 41:157–183
79. Jiang Z, Zuo L, Mingcheng E (2014) Study on multi-objective flexible job-shop scheduling problem considering energy consumption. J Ind Eng Manage 7:589–604