Buddala, Raviteja; Mahapatra, Siba Sankar

Article

An integrated approach for scheduling flexible job-shop using teaching-learning-based optimization method

Journal of Industrial Engineering International

Provided in Cooperation with:
Islamic Azad University (IAU), Tehran

Suggested Citation: Buddala, Raviteja; Mahapatra, Siba Sankar (2019) : An integrated approach for scheduling flexible job-shop using teaching-learning-based optimization method, Journal of Industrial Engineering International, ISSN 2251-712X, Springer, Heidelberg, Vol. 15, Iss. 1, pp. 181-192, https://doi.org/10.1007/s40092-018-0280-8

This Version is available at:
https://hdl.handle.net/10419/267614

Journal of Industrial Engineering International (2019) 15:181–192
https://doi.org/10.1007/s40092-018-0280-8

ORIGINAL RESEARCH

An integrated approach for scheduling flexible job-shop using teaching–learning-based optimization method

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

Received: 7 June 2017 / Accepted: 12 June 2018 / Published online: 18 June 2018

Abstract
In this paper, teaching–learning-based optimization (TLBO) is proposed to solve flexible job shop scheduling problem (FJSP) based on the integrated approach with an objective to minimize makespan. An FJSP is an extension of basic job-shop scheduling problem. There are two sub problems in FJSP. They are routing problem and sequencing problem. If both the sub problems are solved simultaneously, then the FJSP comes under integrated approach. Otherwise, it becomes a hierarchical approach. Very less research has been done in the past on FJSP problem as it is an NP-hard (non-deterministic polynomial time hard) problem and very difficult to solve till date. Further, very less focus has been given to solve the FJSP using an integrated approach. So an attempt has been made to solve FJSP based on integrated approach using TLBO. Teaching–learning-based optimization is a meta-heuristic algorithm which does not have any algorithm-specific parameters that are to be tuned in comparison to other meta-heuristics. Therefore, it can be considered as an efficient algorithm. As best student of the class is considered as teacher, after few iterations all the students learn and reach the same knowledge level, due to which there is a loss in diversity in the population. So, like many meta-heuristics, TLBO also has a tendency to get trapped at the local optimum. To avoid this limitation, a new local search technique followed by a mutation strategy (from genetic algorithm) is incorporated to TLBO to improve the quality of the solution and to maintain diversity, respectively, in the population. Tests have been carried out on all Kacem's instances and Brandimarte's data instances to calculate makespan. Results show that TLBO outperformed many other algorithms and can be a competitive method for solving the FJSP.

Keywords Flexible job shop scheduling · Local search · Makespan · Meta-heuristics · Teaching–learning-based optimization

Introduction

Problems of scheduling occur in many economic domains like airplane scheduling, train scheduling, time table scheduling and especially in the shop scheduling of manufacturing organizations. Scheduling is a crucial factor to improve the productivity of any organization and to meet the deadlines (due dates). Effective scheduling has become mandatory to the manufacturing organizations to maintain the good will among their customers and for survival in the market. Among all the shop scheduling problems, FJSP is the most difficult NP-hard (non-deterministic polynomial time hard) problem till date as pointed out by Garey et al. (1976). The completion time of the last job that leaves the shop floor is defined as the makespan for a scheduling problem by Pinedo (2008). A NP-hard problem is a kind of problem that even a small change in problem size results in exponential increase of computation time. An FJSP consists of all the complexities involved in solving a basic JSP like sequencing problem (i.e., arrangement of operations allotted to a given machine) and additionally, it has routing problem (i.e., each operation's allocation to a machine from given set of machines). The method of solving routing problem first and then the sequencing problem is known as hierarchical approach. Most of the researchers have chosen

✉ Raviteja Buddala
raviteja317@gmail.com

Siba Sankar Mahapatra
mahapatrass2003@gmail.com

[1] Department of Mechanical Engineering, National Institute of Technology Rourkela, Rourkela, Odisha 769008, India

hierarchical approach to solve FJSP. Because, encoding a meta-heuristic to FJSP is easier in hierarchical approach than an integrated approach. But the integrated approach is more powerful as it reduces the computational burden because both the sub-problems of FJSP are tackled simultaneously. So an effort is put in this paper to apply proposed teaching–learning-based optimization (TLBO) in an integrated approach. A recent work by Garmdare et al. (2017) best demonstrates the integrated approach in scheduling problem. TLBO is proposed by Rao et al. (2011) has been applied to different kinds of optimization problems in the past. TLBO is found to be one of the efficient algorithms to give good quality solutions in a reasonable computation time. TLBO has been applied to different scheduling problems like permutation flow shop scheduling by Xie et al. (2014), for job shop scheduling by Keesari and Rao (2014), for flow shop and job shop scheduling cases by Baykasoglu et al. (2014), for flexible job shop scheduling with fuzzy processing times by Xu et al. (2015), for re-entrant flexible flow shop by Shen et al. (2016) and for different scheduling problems by Buddala and Mahapatra (2016, 2017) and it is found that TLBO is one of the efficient meta-heuristic that can be applied to these scheduling problems. So, research is focused in this paper to apply TLBO to solve the NP-hard FJSP.

An effort to solve FJSP is first done by Brucker and Schlie (1990). They solved an FJSP problem with two jobs using a polynomial algorithm. Due to its complexity of FJSP, no exact method has been proposed to solve all sizes of FJSP problems till date. It is not always possible to obtain near optimal solutions to such an NP-hard problem in a reasonable computation time. Therefore, many heuristic and meta-heuristic techniques have been proposed in the recent past to generate near optimal solutions. Brandimarte (1993), based on integrated approach, has applied dispatching rules to solve the routing problem and used tabu search (TS) to solve sequencing problem. The most used algorithm to solve FJSP is genetic algorithm (GA). Chen et al. (1999) have applied GA for the first time, in hierarchical approach using chromosomal representation. Pezzella et al. (2008) have applied GA by integrating several rules to generate initial population, selection and crossover parameters. Gao et al. (2008) have applied GA with advanced mutation and crossover operators and hybridized GA with variable neighborhood descent (VND) technique to increase the search ability of FJSP. Zhang et al. (2011) proposed an improved chromosome representation for GA with a specially designed global selection and local selection parameters to generate good quality solutions. Chang et al. (2015) have proposed a hybrid taguchi GA to solve FJSP. It is a method that encodes feasible solutions to the initial chromosomes developed with Taguchi method behind mating. This is to increase the effectiveness of GA and to increase solution quality. Gambardella and Mastrolilli (1996) have developed two neighborhood functions using local search techniques and proposed two hybrid TS algorithms to solve single objective and multi objective FJSP, respectively. Kacem et al. (2002a) have proposed a pareto-optimal approach by hybridization of evolutionary algorithms and fuzzy logic to solve multi objective FJSP. Kacem et al. (2002b) have proposed two approaches called approach by localization and evolutionary approach to solve FJSP with makespan criterion in hierarchical approach. Based on the behavior of flying birds (the swarm intelligence), Xia and Wu (2005) have applied particle swarm optimization (PSO). By hybridizing PSO with the local search algorithm called simulated annealing (SA) to solve multi-objective FJSP. Singh and Mahapatra (2012) and Singh and Mahapatra (2016) have proposed PSO and quantum behaved particle swarm optimization (QPSO), respectively, using chaotic numbers generated from logistic mapping function instead of random numbers to solve FJSP. Fattahi et al. (2007) have proposed a mathematical model and used different heuristic techniques to solve FJSP. Six different hybrid heuristics were proposed and results show that hybrid algorithm combination of TS and SA outperformed other heuristic combinations. Garmsiri and Abassi (2012) and Liouane et al. (2007) have used a combination of ant system (AS) algorithm with TS algorithm. They proposed a hybrid ant colony optimization (HACO) to solve six FJSP problems where TS is used as a local search algorithm. Xing et al. (2010) have proposed a knowledge-based ant colony approach (KBACO) by integrating the knowledge model to ant colony optimization (ACO) to solve FJSP. Xing et al. (2009a) have proposed a new simulation model to solve multi-objective FJSP. Xing et al. (2009b) have proposed a new algorithm based on some empirical knowledge to solve multi-objective FJSP. Bagheri et al. (2010) have applied an artificial immune algorithm (AIA) to solve FJSP. Li et al. (2010) have proposed a hybrid tabu search algorithm (HTSA) to solve multi-objective FJSP where a variable neighborhood structure with two adaptive rules is used to hybridize TS to increase its local search ability. Based on the application of multiple independent searches that increase the exploration of search space, Yazdani et al. (2010) have proposed a parallel variable neighborhood search (PVNS) technique. This technique uses various neighborhood techniques to solve FJSP. Based on the natural phenomenon of bee colony, Wang et al. (2012) have proposed an artificial bee colony (ABC) algorithm to solve FJSP with makespan criterion. Li et al. (2014) have proposed a discrete artificial bee colony (DABC) to solve FJSP considering the preventive maintenance and non-preventive maintenance conditions. Based on the migration strategy of animals from one place to

another, Rahmati and Zandieh (2012) have proposed a biogeography-based optimization (BBO) to solve FJSP. Yuan et al. (2013) have proposed a hybrid harmony search (HHS) algorithm based on an integrated approach. Some local search technique is embedded in the harmony search algorithm to solve FJSP. Using some existing heuristics applied to harmony search (HS), Gao et al. (2016) have proposed a discrete harmony search (DHS) to solve multiobjective FJSP. Karthikeyan et al. (2015) have proposed a hybrid discrete firefly algorithm (HDFA) to solve multiobjective FJSP. Maleki-Darounkolaei et al. (2012), Noori-Darvish and Tavakkoli-Moghaddam (2012), Mirabi et al. (2014) and Kia et al. (2017) worked on sequence-dependent scheduling problems. Nouri et al. (2017) proposed a holonic multiagent model to solve FJSP.

Problem formulation

As JSP itself is an NP-hard, its extension FJSP is also an NP-hard. To find an optimal solution to an NP-hard problem in a reasonable computation time, is not always possible. Therefore, it is advisable to find a near optimal solution using meta-heuristic techniques in a reasonable computation time than searching an optimal solution with great computational effort. In a flexible job-shop scheduling problem, n jobs are to be arranged on m machines such that all the operations are executed successfully. There are n jobs (i = 1, 2, 3, …, n) with each job i having a predetermined number of operations J (j = 1, 2, 3, …, J) in a sequence that are to be executed on m machines (k = 1, 2, 3, … m). Manne (1960) proposed FJSP in a mixed integer linear programming (MILP) formulation with the following assumptions. All machines are different from each other. All jobs are different from each other. Machine breakdown during the operations is not allowed (non-preemption condition, i.e., an operation without any interruption must be completed). Any machine can perform at most one operation at an instant (resource constraint).

The following are the notations used for MILP formulation:

m	Total number of machines
n	Total number of jobs
Oij	ith job's jth operation
Ji	ith job's total number of operations
$Pijk$	Oij operation's processing time if performed on machine k
B	A big number
Mij	Number of machines available for processing operation Oij
$Sijk$	Begin time of operation Oij on machine k
$Cijk$	Completion time of operation Oij on machine k
Ci	Final completion time of job i in shop floor
Cm	Makespan
i, h	Job index (1, 2, 3, …, n)
j, g	Operation index (1, 2, 3, …, Ji)
k	Machine index (1, 2, 3, …, m)
$Xijk$	1, if Oij is executed on machine k 0, otherwise
$Zijhgk$	1, if operation Ohg succeeds operation Oij on machine k 0, otherwise

The objective is to minimize makespan

$$Cm \geq Ci \quad \forall i \tag{1}$$

Subject to constraints

$$Ci \geq \sum Cijk \quad \forall k \in Mij \quad \& \quad \forall i,j \in Ji \tag{2}$$

$$Sijk + Cijk \leq Xijk * B \quad \forall k \in Mij \quad \& \quad \forall i,j \in Ji \tag{3}$$

$$Cijk - Sijk \geq Pijk - (1 - Xijk) * B \quad \forall k \in Mij \quad \& \quad \forall i,j \in Ji \tag{4}$$

$$Sijk - Cjgk \geq (Zijhgk) * B \quad \forall i \leq h, \quad \forall j \quad \& \quad k \in Mij \cap Mhg \tag{5}$$

$$Sijk - Cjgk \geq (1 - Zijhgk) * B \quad \forall i \leq h, \quad \forall j \quad \& \quad k \in Mij \cap Mhg \tag{6}$$

$$\sum Sijk - \sum C(ij-1)k \geq 0 \quad \forall i, \quad \forall j = (2, 3, \ldots, Ji) \quad \& \quad \forall k \in Mij \tag{7}$$

$$\sum Xijk = 1 \quad \forall i,j \quad \& \quad k \in Mij \tag{8}$$

$$Sijk \geq 0 \quad \& \quad Cijk \geq 0 \quad \forall i,j,k \tag{9}$$

$$Cm \geq Ci \quad \forall i \tag{10}$$

Inequality (1) determines the objective makespan. Constraint (2) governs completion time of all the jobs. Constraints (3) and (4) restrict that the minimum time difference between starting and completion times must be processing time on that particular machine k. In the set $Mij \cap Mhg$, constraints (5) and (6) assure that resource constraint is not violated (i.e., no two operations are allotted to same machine at a given time). Precedence relationships between them are ensured by the constraint (7), i.e., second operation of a job cannot be started until the first operation of the same job is completed. An operation can be executed on one and only one machine. This is assured by the constraint (8). If an operation is not assigned to a machine k, then the starting and completion times of that operation are zero on that machine k. This is ensured by constraint (9). Constraint (10) determines the makespan.

As it is not always possible to obtain near optimal solutions in a reasonable computation time, scheduling of

jobs in FJSP is acknowledged as NP-hard problem is explained by Lenstra et al. (1977). The exact methods like dynamic programming proposed by Potts and Van Wassenhove (1987), branch and bound proposed by Brucker et al. (1994) and Artigues and Feillet (2008) are capable of solving only small size problems due to large complexity involved in FJSP. Most of them fail to solve large size problems to obtain good results and they also require large computation time and huge memory. The real world FJSP problems of medium and large-scale size are beyond the scope of these exact methods, in spite of their relative success rate to achieve optimal solution. So researchers focused on non-exact heuristic methods like dispatching rules. This research further continued to the application of meta-heuristic techniques like GA, PSO, TS, SA, ABC, BBO, HS, etc. which take less time compared to heuristic techniques.

Teaching–learning-based optimization

TLBO is proposed by Rao et al. (2011), Rao and Patel (2012, 2013) with an inspiration from the general teaching–learning process that how a teacher influences the knowledge of students. Students and teacher are the two main objects of a class and the algorithm explains the two modes of learning, i.e., via teacher (teacher phase) and discussion among the fellow students (student phase). A group of students of the class constitute the population of the algorithm. In any iteration, the best student of the class becomes the teacher. Execution of the TLBO is explained in two phases: they are teacher phase and student phase.

Teacher phase

A teacher puts his/her best to increase the knowledge of his/her students to his level. But practically it is not possible as learning by a student depends on his/her caliber to learn. In this process, each student's knowledge increases and in turn the average knowledge of the class increases. At any iteration, let Z_{mean} denote the mean knowledge of the class and teacher of the class is denoted as Z_{teacher}. Then, the increased knowledge of a student is given by the expression

$$Z_{\text{new } i} = Z_{\text{old } i} + r \times (Z_{\text{teacher}} - (T_{\text{f}} \times Z_{\text{mean}})) \tag{11}$$

where 'i' denotes student of the class, 'r' is a random number between zero to one and 'T_{f}' is called the teaching factor whose value is randomly chosen as one or two and there is no tuning of this teaching factor even though 'T_{f}' is an algorithm-specific parameter of TLBO.

Student phase

After a class is taught, students discuss among themselves. In this process, the knowledge of all students' increases. At any iteration, let Z_a and Z_b be two students who discuss after the class, $a \neq b$. Then, the increased knowledge of the student is given by the expression

$$Z_{\text{new } a} = Z_{\text{old } a} + r_i \times (Z_a - Z_b) \text{ if } F(Z_a) \leq F(Z_b) \tag{12}$$

$$Z_{\text{new } a} = Z_{\text{old } a} + r_i \times (Z_b - Z_a) \text{ if } F(Z_b) < F(Z_a) \tag{13}$$

Accept $Z_{\text{new } i}$ if it gives a better function value.

Problem mapping and representation

A real number encoding system is proposed in this paper which is used by Niu et al. (2009) to solve hybrid flow shop scheduling. To allot each operation of a job to a machine, integer part of the real number is used and to sequence the operations on each machine, fractional part is used. For better understanding, consider the example problem given in Table 1. Using the processing times data from Table 1, a priority order matrix of machines in the increasing order of

Table 1 Example problem

Job	Operation	Machine 1	Machine 2	Machine 3	Machine 4
1	O11	5	3	1	2
	O12	2	4	6	3
2	O21	3	5	7	2
	O22	1	2	3	4
	O23	3	2	4	6
3	O31	5	1	4	2
	O32	6	4	5	3
	O33	5	2	6	4
	O34	4	3	2	1

Table 2 Order of priority

Job	Operation	Priority 1	Priority 2	Priority 3	Priority 4
1	O11	3	4	2	1
	O12	1	4	2	3
2	O21	4	1	2	3
	O22	1	2	3	4
	O23	2	1	3	4
3	O31	2	4	3	1
	O32	4	2	3	1
	O33	2	4	1	3
	O34	4	3	2	1

Table 3 Stochastic representation of subject value

Operation	O11	O12	O21	O22	O23	O31	O32	O33	O34
Value of subject	2.8430	1.4356	1.1724	2.5219	3.2456	1.7951	3.5842	1.3216	4.2427
Priority	2	1	1	2	3	1	3	1	4
Machine assigned	4	1	4	2	3	2	3	2	1

processing times for each operation is calculated as shown in the Table 2.

If there is a tie in processing times, machine with lower index number is given priority. The integer part is used to allot a machine from this priority order matrix to an operation. The stochastic representation of subject value is explained in Table 3. The initial subject values of student population are generated at random. Each subject value should be a positive real number. The maximum value a subject can have is the total number of machines available (tma). The minimum possible number that can be used is 1. So subject values are generated between the range (1, 1 + tma). For example, in Table 3 the operation O11 has a subject value of 2.8430 with integer value 2. So the operation O11 is allotted to second priority machine 4 according to the priority order matrix from Table 2. After the allotment, operations on each machine are sequenced in the increasing order of fractional values. For example, operations O12 and O34 are allotted to machine 1. The fractional value of O34 is less than O12. Therefore, O34 is sequenced first and O12 later. In this way, an initial sequencing is done (in Table 4) on the machines according to real number encoding system proposed by Niu et al. (2009).

Proposed algorithm

1. Input all the problem data like number of jobs, number of operations of each job, number of machines available for each job and the corresponding processing times.
2. Initialize students of the class. Generate initial subject values of the students randomly within the range. $Sij = Sij\ \min + (Sij\ \max - Sij\ \min) \times r$ where Sij is the subject value of jth operation of job i. $Sij\ \min = 1$, $Sij\ \max = (1 + \text{tma})$ and r is a random number between (0, 1).
3. Generate the schedule using encoding scheme as proposed in problem mapping representation and the method to eliminate infeasible solution.
4. Evaluate each student's knowledge (makespan).
5. Now evaluate the mean knowledge of all the students in the class.
6. Update the knowledge of all the students with teacher phase and student phase (Eqs. 11, 12 and 13) of TLBO.
7. Apply the local search as explained in the local search section.
8. Identify the best student of the class and replace the earlier teacher with best student if best student fitness is better than teacher.
9. Apply the mutation technique for every five iterations by randomly replacing 3% of the population.
10. Repeat the cycle to step 3 until the termination criterion is met.
11. End.

Table 4 Initial sequence of operations before optimization

Machine 1	O34	O12	
Machine 2	O33	O22	O31
Machine 3	O23	O32	
Machine 4	O21	O11	

Method to eliminate infeasible solution

Because of the complexity of FJSP, there is very good chance for infeasible solution generation by the population of an algorithm during the run time. In this process too for example, on machine 2 in Table 4, operation O33 is sequenced before O31 which is an infeasible solution. In this paper, we propose a new method to eliminate infeasible solution when a real number encoding system is used. If we observe clearly, this problem arises when two or more operations of a job are allotted to a same machine and later operations of that job have low fractional value than the preceding operations. Here, O33 has a fractional value 0.3216 which is less than O31 fractional value 0.7951. Due to this reason, O33 is allotted before O31.

To eliminate this problem, all the fractional values of each job are first arranged in an ascending order as prescribed in Table 5. For example, job 1 operations O11 and O12 have fractional values 0.8430 and 0.4356, respectively. After the ascending order arrangement, the final fractional values of O11 and O12 are 0.4356 and 0.8430, respectively. In this way, we can eliminate the possibility of infeasible solution generation. The pseudo-code for the same is given in Fig. 1. The initial gantt chart solution to

the example problem before the application of TLBO is given in Fig. 2.

Local search (LS)

Like many meta-heuristics, TLBO also requires a local search technique to improve the solution exploration capacity of the algorithm while solving FJSP. A new local search technique is proposed in this paper. This local search technique can be applied to any meta-heuristic technique in the future. Only solutions with best makespan are considered for local search at every iteration. This method is explained in two steps. (1) Sequence swap (2) Machine swap. Before going to these two steps, all the critical operations of the solution are to be found. To improve the solution quality of FJSP, a promising critical path concept in operation scheduling phase (Zhang et al. 2007) is used as explained below.

Sequence swap

Out of these critical operations, only operations allotted to the same machine such that two or more operations which are adjacent to each other are collected and pairwise swapping (pair exchange method used by Xia and Wu 2005) is done to these collected operations and makespan is evaluated for each swap. If the swapping gives a better makespan value, then the old solution is replaced with the current new solution. For example, in Fig. 2, operations marked with a black line are critical operations. Operations O11 and O21 on machine 4 and operations O22 and O33 on machine 2 come under this category. These operations are swapped and checked for new makespan.

Machine swap

After sequence swap, each of the critical operation is picked and is allotted to its other machines from priority order one to three and the fitness value is evaluated. Generally, to obtain a minimum makespan, operations are to be allotted to machines that process in less time. So machine change technique is limited to a maximum of first three priority machines. In Fig. 2, each of the above-mentioned critical operations is allotted to their first three priority machines and

```
Let k=0 and l=0;
For i= 1 to number of jobs
        For j= 1 to maximum operation number in job(i)
                mat1(i,j)=x(k+j);
        end
        k=k+ maximum operation number in job(i);
end
mat2= sorted matrix of mat1 along row wise;
For i= 1 to number of jobs
        For j= 1 to maximum operation number in job(i)
                x(l+j)=mat2(i,j);
        end
        l=l+ maximum operation number in job(i);
end
result=x;
```

Fig. 1 Pseudocode to eliminate infeasible solution

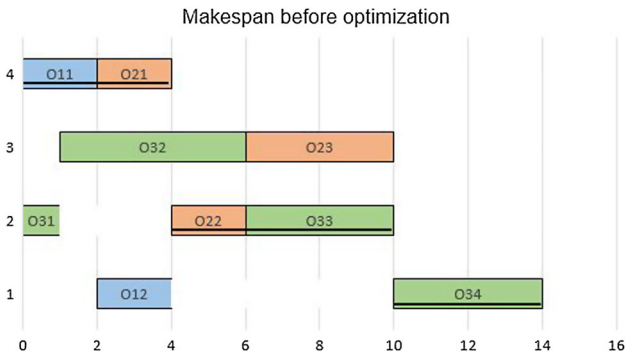

Fig. 2 Initial solution as per final subject values from Table 5

checked for makespan. This is done by changing the integer value of that particular subject value. For example, O11 has a subject value of 2.4356. The integer value of 2 is replaced by 1 and then 3, which makes the subject value as 1.4356 and 3.4356, respectively. In this way, after the machine exchange process for each critical operation, if the obtained makespan value is better, then the solution is replaced with the current new better solution. Table 6 shows the final sequence of operations before optimization.

Mutation strategy (MS)

A sudden change in the gene of the offspring compared to parent gene during the natural evolution process is called mutation. Even though exploration capability of TLBO

Table 5 Elimination of infeasible solutions

Operation	O11	O12	O21	O22	O23	O31	O32	O33	O34
Value of subject	2.8430	1.4356	1.1724	2.5219	3.2456	1.7951	3.5842	1.3216	4.2427
Fractional value	0.8430	0.4356	0.1724	0.5219	0.2456	0.7951	0.5842	0.3216	0.2427
Final fractional value	0.4356	0.8430	0.1724	0.2456	0.5219	0.2427	0.3216	0.5842	0.7951
Final value of subject	2.4356	1.8430	1.1724	2.2456	3.5219	1.2427	3.3216	1.5842	4.7951

improved with the local search technique, we observe that TLBO gets trapped at the local optimum. Observing the makespan values of all the students, it is clear that almost all the students reach the same knowledge level (local optimum) after several iterations and there is no further improvement. To eliminate this drawback and to maintain diversity in the population, thus increasing the balance between exploration and exploitation, mutation technique from the genetic algorithm is incorporated to the algorithm. The total population of the class considered here is 100. Out of the total population, randomly two percent of population is made to undergo mutation for every five iteration. It means that two percent population is replaced with a new random solution using the equation in step 2 of proposed algorithm. The reasons to implement mutation strategy are (1) there is an improvement in the quality of solutions obtained and (2) this does not increase the computation burden much.

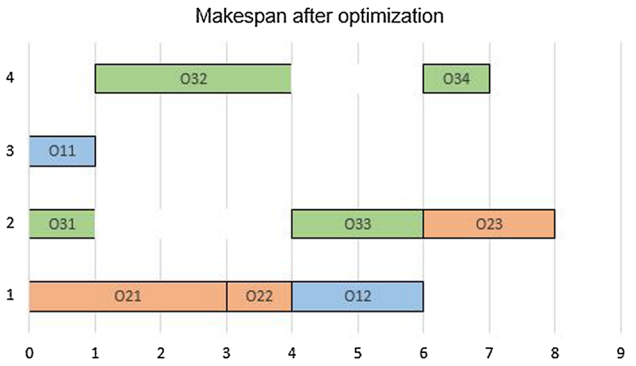

Fig. 3 Final solution obtained from the proposed algorithm

Table 7 Final sequence of operations of optimized solution

Machine 1	O21	O22	O12
Machine 2	O31	O33	O23
Machine 3	O11		
Machine 4	O32	O34	

Results and discussion

The computational experiments aim to find the performance of TLBO to solve FJSP with makespan as the objective. Figure 3 shows the final solution obtained for the example problem in gantt chart and Table 7 shows the final sequence of operations for obtained optimized solution of the example problem using the proposed algorithm. We can observe that solution improved from a makespan value of 14 time units to eight time units. The experiments are conducted using MATLAB software on an i7 processor running at 3.40 GHz on the Windows 7 operating system. Tests have been carried out on all the instances by Brandimarte (1993) and Kacem et al. (2002a, b). There are five Kacem's instances with problem size ranging from 4×5 to 15×10. There are a set of 10 Brandimarte's problems with size ranging from 10×6 to 20×15. These are the only two data sets that are widely solved and available for comparison from the literature.

When the teaching–learning-based optimization alone is applied to solve all the problems, we observed that TLBO is able to attain the best makespan values to four out of five Kacem's problems. From this, we can conclude that TLBO alone is sufficient to solve the small and medium size problems. For the large size problems, i.e., fifth Kacem's problem and all Brandimarte's ten problems (MK01–MK10), TLBO alone could not obtain the best makespan values. So there is a limitation in TLBO to solve large problems. To improve the efficiency of TLBO with an aim to solve large problems, the local search technique proposed in this paper is applied and once again makespan values of all the problems are calculated. This time we found a very good improvement in the quality of solutions obtained. Equation 14 gives the percentage improvement (PI) of each problem and Eq. 15 gives the average percentage improvement (API) of all the problems, where 'X' is the index of the problem and 'N' is the total number of problems. There is an improvement in all the eleven big problems and the percentage improvement (PI) values are tabulated in fifth column of Table 8. The average percentage improvement (API) of proposed local search is 9.771.

$$\mathrm{PI} = \frac{\mathrm{TLBO} - \mathrm{Improved\ TLBO}}{\mathrm{TLBO}} \times 100 \tag{14}$$

$$\mathrm{API} = \frac{\sum_{X=1}^{N} \mathrm{PI}(X)}{N} \tag{15}$$

As best student of the class is considered as teacher in TLBO, after few iterations all the students learn and reach the same knowledge level. Due to this reason, it is observed that there is a loss of diversity in the population. So, like many meta-heuristics, TLBO also has a tendency to get trapped at the local optimum. To avoid this limitation, mutation strategy from genetic algorithm is incorporated to improve the quality of the solution and to maintain diversity. Once again tests have been carried out to find the makespan value of all the problems. Now, we observed not only a very good improvement in the quality of solutions but also four more best solutions are obtained, i.e., for the

Table 6 Final sequence of operations before optimization

Machine 1	O12	O34	
Machine 2	O31	O22	O33
Machine 3	O32	O23	
Machine 4	O11	O21	

Table 8 Percentage improvement in solutions

SI. no.	Problem size	TLBO	TLBO + LS	PI	TLBO + LS + MS (proposed TLBO)	PI
1	4 × 5	**11**	**11**	0	**11**	0
2	8 × 8	**14**	**14**	0	**14**	0
3	10 × 7	**11**	**11**	0	**11**	0
4	10 × 10	**7**	**7**	0	**7**	0
5	15 × 10	14	13	7.142	12	14.286
6	10 × 6	45	42	6.667	**40**	11.111
MK01						
7	10 × 6	33	30	9.090	28	15.152
MK02						
8	15 × 8	285	222	22.105	**204**	28.421
MK03						
9	15 × 8	97	74	23.711	63	35.052
MK04						
10	15 × 4	187	176	5.882	**172**	8.0214
MK05						
11	10 × 15	94	80	14.893	65	30.851
MK06						
12	20 × 5	166	150	9.638	144	13.253
MK07						
13	20 × 10	578	532	7.958	**523**	9.515
MK08						
14	20 × 10	394	332	15.736	311	21.066
MK09						
15	20 × 15	337	257	23.738	214	36.499
MK10						
Average percentage improvement (API)				**9.771**		**14.882**

Numbers in bold indicate the best values

problems of MK01, MK03, MK05 and MK08. For Kacem's fifth problem and MK02 problems, the second best solutions are obtained. The percentage improvement of proposed algorithm is tabulated in the last column of Table 8. The API value is 14.882 which is considerably a large improvement.

Table 9 shows the comparison of results obtained by TLBO with the previously solved other meta-heuristics. Out of 15 problems solved, the proposed TLBO gives the best solutions for eight problems. Comparing the TLBO with other algorithms, it outperformed genetic algorithm proposed by Chen et al. (1999) in four problems. Out of the three problems solved by Xia and Wu (2005) using PSO + SA from Kacem's instances, TLBO gives better result to one problem than PSO + SA. TLBO outperformed HACO, used by Liouane et al. (2007), in one problem and gives equal results to all other five problems. TLBO outperformed Xing's algorithm (used by Xing et al. 2009b) in six problems. TLBO outperformed AIA (used by Bagheri et al. 2010) in two problems. TLBO also outperformed BBO (used by Rahmati and Zandieh 2012) in four problems. Although other algorithms like ABC and DABC (used by Wang et al. 2012; Li et al. 2014) and HHS and DHS (used by Yuan et al. 2013; Gao et al. 2016) outperformed TLBO in few problems, the algorithms seem to be complex and contain algorithm-specific parameters to be tuned to generate the best solutions. On the other hand, TLBO is a simple algorithm and does not possess any algorithm-specific tuning parameters. Artificial bee colony (ABC) uses the number of scout bees, onlooker bees and employed bees, whereas Harmony search (HS) uses pitch adjusting rate, harmony memory consideration rate and number of improvisations. As tuning the parameters is a difficult task, this drawback can be eliminated using TLBO. Unlike other algorithms, parameter-less feature of TLBO eliminates the drawback of solving the NP-hard FJSP problem again and again until the right kind of tuning parameters is found. From these results, it is clear that TLBO is a competitive and simple algorithm that can be applied to solve FJSP.

Figures 4 and 5 show the rate of convergence of TLBO towards the optimal solution. After conducting experiments

Table 9 Results comparison of 15 benchmark problems of FJSP

SI. no.	Size	GA Chen et al. (1999)	PSO + SA Xia and Wu (2005)	HACO Liouane et al. (2007)	Xing's algorithm Xing et al. (2009b)	AIA Bagheri et al. (2010)	HTSA Li et al. (2010)	ABC Wang et al. (2012)	BBO Rahmati and Zandieh (2012)	HHS Yuan et al. (2013)	DABC Li et al. (2014)	HDFA Karthikeyan et al. (2015)	DHS Gao et al. (2016)	Proposed TLBO
1	4 × 5	NA	NA	**11**	**11**	NA	**11**	**11**	**11**	NA	NA	**11**	NA	**11**
2	8 × 8	16	15	NA	**14**	**14**	**14**	**14**	**14**	**14**	NA	**14**	NA	**14**
3	10 × 7	NA	NA	**11**	**11**	NA	**11**	**11**	NA	NA	NA	**11**	NA	**11**
4	10 × 10	**7**	**7**	**7**	**7**	**7**	**7**	**7**	**7**	**7**	NA	**7**	NA	**7**
5	15 × 10	NA	12	12	**11**	**11**	**11**	**11**	12	**11**	NA	**11**	NA	12
6 MK01	10 × 6	**40**	NA	NA	42	**40**	**40**	**40**	**40**	**40**	**40**	NA	**40**	**40**
7 MK02	10 × 6	29	NA	28	28	**26**	**26**	**26**	28	**26**	**26**	NA	28	28
8 MK03	15 × 8	**204**	NA	NA	**204**	**204**	**204**	**204**	**204**	**204**	**204**	NA	**204**	**204**
9 MK04	15 × 8	63	NA	NA	68	**60**	61	**60**	64	**60**	**60**	NA	**60**	63
10 MK05	15 × 4	181	NA	NA	177	173	**172**	**172**	173	**172**	**172**	NA	**172**	**172**
11 MK06	10 × 15	60	NA	68	75	63	65	60	66	**58**	NA	NA	67	65
12 MK07	20 × 5	148	NA	NA	150	140	140	**139**	144	**139**	**139**	NA	143	144
13 MK08	20 × 10	**523**	NA	NA	**523**	**523**	**523**	**523**	**523**	**523**	**523**	NA	**523**	**523**
14 MK09	20 × 10	308	NA	NA	311	312	310	**307**	310	**307**	NA	NA	309	311
15 MK10	20 × 15	212	NA	NA	227	214	214	208	230	**205**	NA	NA	212	214

Numbers in bold indicate the best values

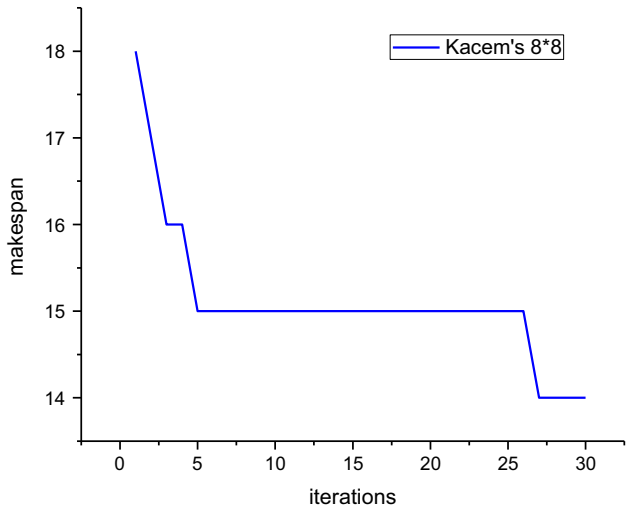

Fig. 4 Rate of convergence of TLBO for Kacem's 8 by 8 problem

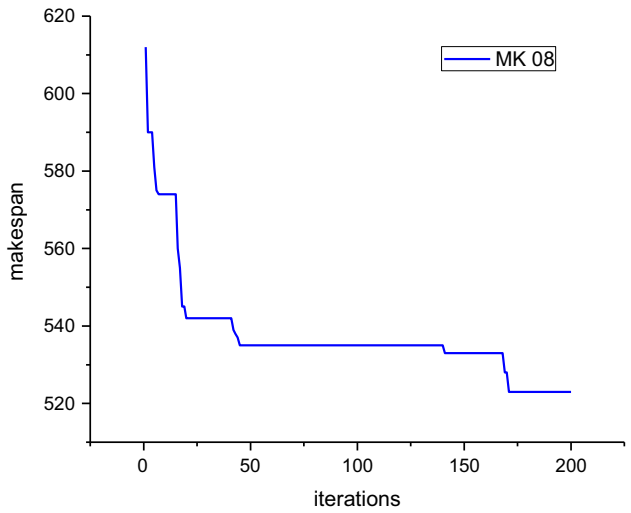

Fig. 5 Rate of convergence of TLBO for MK08 problem

for different mutation percentage from the range one to five, it is found that most of the problems gave good results for two percent mutation. So mutation percent is fixed to two. Figure 6 shows the variation of results (makespan) of the problems for different mutation percentages.

Conclusion

In this paper, one of the most difficult NP-hard flexible job shop scheduling problems with makespan as criterion is considered and an efficient and effective teaching–learning-based optimization is used to generate near optimal schedules for fifteen benchmark problems. A new local search procedure has been proposed and it is found to be effective. A new technique to successfully overcome the

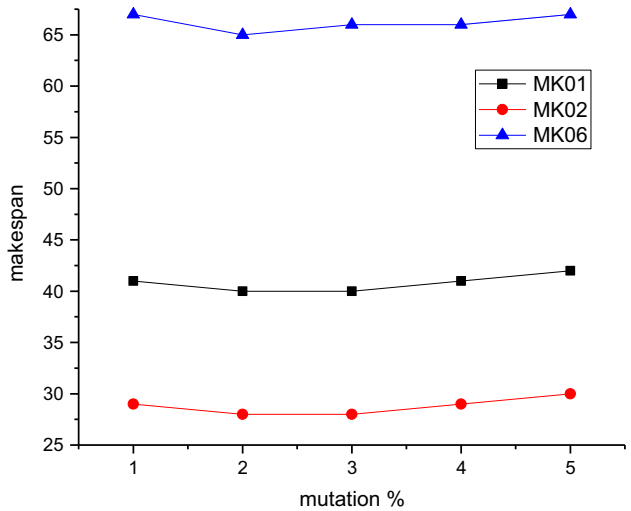

Fig. 6 Result variation with mutation percentage

infeasible solutions that are generated during the run time is proposed using the real number encoding system. The mutation technique from the genetic algorithm is incorporated to algorithm to maintain the diversity in the population. Results show that the proposed TLBO is found to be one of the good problem solving approaches for solving FJSP, as it out performed many other algorithms from the literature. It gave the best results to 8 problems out of 15. As tuning the parameters of meta-heuristics itself is a herculean task, this paper stands as a basement for future research to concentrate or develop tuning parameter less algorithms to solve FJSP. The proposed TLBO avoids this drawback and, thus, reduces the computational burden too. The future work can be extended to hybridize TLBO with other local search techniques. Also, a hierarchical approach-based TLBO can be experimented to solve FJSP. The work can be further extended by considering the various uncertainties that are encountered in a real-life FJSP problem. Also, a multi-objective optimization study of FJSP can be carried out in the future.

References

Artigues C, Feillet D (2008) A branch and bound method for the job-shop problem with sequence-dependent setup times. Ann Oper Res 159:135–159

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

Baykasoglu A, Hamzadayi A, Kose 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

Brandimarte P (1993) Routing and scheduling in a flexible job shop by tabu search. Ann Oper Res 41:157–183

Brucker P, Schlie R (1990) Job-shop scheduling with multi-purpose machines. Computing 45:369–375

Brucker P, Jurisch B, Sievers B (1994) A branch and bound algorithm for the job-shop scheduling problem. Discrete Appl. Math. 49:107–127

Buddala R, Mahapatra SS (2016) An effective teaching learning based optimization for flexible job shop scheduling. In: International conference on electrical, electronics, and optimization techniques (ICEEOT). IEEE, pp 3087–3092. https://doi.org/10.1109/ICEEOT.2016.7755269

Buddala R, Mahapatra SS (2017) Improved teaching–learning-based and JAYA optimization algorithms for solving flexible flow shop scheduling problems. J Ind Eng Int. https://doi.org/10.1007/s40092-017-0244-4

Chang HC, Chen YP, Liu TK, Chou JH (2015) Solving the flexible job shop scheduling problem with makespan optimization by using a hybrid taguchi-genetic algorithm. IEEE Access 3:1740–1754

Chen H, Ihlow J, Lehmann C (1999) A genetic algorithm for flexible job-shop scheduling. In: 1999 IEEE international conference on robotics and automation, 1999. Proceedings, vol 2. IEEE, pp 1120–1125

Fattahi P, Mehrabad MS, Jolai F (2007) Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. J Intell Manuf 18:331–342

Gambardella LM, Mastrolilli M (1996) Effective neighborhood functions for the flexible job shop problem. J Sched 3:3–20

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

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

Garey MR, Johnson DS, Sethi R (1976) The complexity of flowshop and jobshop scheduling. Math Oper Res 1:117–129

Garmdare HS, Lotfi MM, Honarvar M (2017) Integrated model for pricing, delivery time setting, and scheduling in make-to-order environments. J Ind Eng Int. https://doi.org/10.1007/s40092-017-0205-y

Garmsiri M, Abassi MR (2012) Resource leveling scheduling by an ant colony-based model. J Ind Eng Int 8(1):7

Kacem I, Hammadi S, Borne P (2002a) Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic. Math Comput Simul 60:245–276

Kacem I, Hammadi S, Borne P (2002b) Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. IEEE Trans Syst Man Cybern Part C Appl Rev 32:1–3

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

Keesari HS, Rao RV (2014) Optimization of job shop scheduling problems using teaching–learning-based optimization algorithm. Opsearch 51:545–561

Kia H, Ghodsypour SH, Davoudpour H (2017) New scheduling rules for a dynamic flexible flow line problem with sequence-dependent setup times. J Ind Eng Int. https://doi.org/10.1007/s40092-017-0185-y

Lenstra JK, Kan AR, Brucker P (1977) Complexity of machine scheduling problems. Ann Discrete Math 1:343–362

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

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 Modell 38:1111–1132

Liouane N, Saad I, Hammadi S, Borne P (2007) Ant systems and local search optimization for flexible job shop scheduling production. Int J Comput Commun Control 2:174–184

Maleki-Darounkolaei A, Modiri M, Tavakkoli-Moghaddam R, Seyyedi I (2012) A three-stage assembly flow shop scheduling problem with blocking and sequence-dependent set up times. J Ind Eng Int 8(1):26

Manne AS (1960) On the job-shop scheduling problem. Oper Res 8:219–223

Mirabi M, Ghomi SF, Jolai F (2014) A novel hybrid genetic algorithm to solve the make-to-order sequence-dependent flow-shop scheduling problem. J Ind Eng Int. https://doi.org/10.1007/s40092-014-0057-7

Niu Q, Zhou T, Ma S (2009) A quantum-inspired immune algorithm for hybrid flow shop with makespan criterion. J UCS 15(4):765–785

Noori-Darvish S, Tavakkoli-Moghaddam R (2012) Minimizing the total tardiness and makespan in an open shop scheduling problem with sequence-dependent setup times. J Ind Eng Int 8(1):25

Nouri HE, Driss OB, Ghédira K (2017) Solving the flexible job shop problem by hybrid metaheuristics-based multiagent model. J Ind Eng Int. https://doi.org/10.1007/s40092-017-0204-z

Pezzella F, Morganti G, Ciaschetti G (2008) A genetic algorithm for the flexible job-shop scheduling problem. Comput Oper Res 35:3202–3212

Pinedo M (2008) Scheduling: theory, algorithms and systems, 3rd edn. Springer, Berlin

Potts CN, Van Wassenhove LN (1987) Dynamic programming and decomposition approaches for the single machine total tardiness problem. Eur J Oper Res 32:405–414

Rahmati SH, Zandieh M (2012) A new biogeography-based optimization (BBO) algorithm for the flexible job shop scheduling problem. Int J Adv Manuf Technol 58:1115–1129

Rao R, Patel V (2012) An elitist teaching–learning-based optimization algorithm for solving complex constrained optimization problems. Int J Ind Eng Comput 3:535–560

Rao R, Patel V (2013) Comparative performance of an elitist teaching–learning-based optimization algorithm for solving unconstrained optimization problems. Int J Ind Eng Comput 4:29–50

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

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

Singh MR, Mahapatra SS (2012) A swarm optimization approach for flexible flow shop scheduling with multiprocessor tasks. Int J Adv Manuf Technol 62:267–277

Singh MR, Mahapatra SS (2016) A quantum behaved particle swarm optimization for flexible job shop scheduling. Comput Ind Eng 93:36–44

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 Technol 60:303–315

Xia W, Wu Z (2005) An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. Comput Ind Eng 48:409–425

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

Xing LN, Chen YW, Yang KW (2009a) An efficient search method for multi-objective flexible job shop scheduling problems. J Intell Manuf 20:283–293

Xing LN, Chen YW, Yang KW (2009b) Multi-objective flexible job shop schedule: design and evaluation by simulation modeling. Appl Soft Comput 9:362–376

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

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

Yazdani M, Amiri M, Zandieh M (2010) Flexible job-shop scheduling with parallel variable neighborhood search algorithm. Expert Syst Appl 37:678–687

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

Zhang C, Li P, Guan Z, Rao Y (2007) A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem. Comput Oper Res 34:3229–3242

Zhang G, Gao L, Shi Y (2011) An effective genetic algorithm for the flexible job-shop scheduling problem. Expert Syst Appl 38:3563–3573