Int J Adv Manuf Technol (2015) 79:135–145
DOI 10.1007/s00170-015-6795-x

ORIGINAL ARTICLE

A hybrid computer simulation-adaptive neuro-fuzzy inference system algorithm for optimization of dispatching rule selection in job shop scheduling problems under uncertainty

A. Azadeh · N. Hosseini · S. Abdolhossein Zadeh · F. Jalalvand

Received: 22 August 2014 / Accepted: 8 January 2015 / Published online: 30 January 2015

Abstract Unstable environment of industrial systems is a source of various uncertainties in production features such as processing times. Moreover, selecting appropriate dispatching rules is a complex and significant issue in practical problems under uncertainty. Most previous studies have pointed out that using a single dispatching rule does not necessarily result in an optimal schedule. This study proposes a novel hybrid algorithm based on computer simulation and adaptive neuro-fuzzy inference system (ANFIS) to select optimal dispatching rule for each machine in job shop scheduling problems (JSSPs) under uncertain conditions so that makespan is minimized. It captures uncertainty using fuzzy set theory and assumes that processing times are in the form of fuzzy numbers. This algorithm contributes to the previous works in two important ways. First, the inherent uncertainty of JSSPs is reflected in fuzzy processing times. Second, this is the first study that develops an approach based on computer simulation and ANFIS for selecting the optimal dispatching rules and minimizing the makespan in JSSPs under uncertainty. The computational results demonstrate the superiority of this algorithm over the previous studies in the literature.

Keywords Job shop scheduling problem · Fuzzy processing times · Computer simulation · Adaptive neuro-fuzzy inference system · Makespan minimization · Uncertainty

A. Azadeh (✉) · N. Hosseini · S. Abdolhossein Zadeh
School of Industrial Engineering and Center of Excellence for Intelligent Based Experimental Mechanic, University of Tehran, Tehran, Iran
e-mail: aazadeh@ut.ac.ir

F. Jalalvand
Department of Industrial Engineering, University of Science and Technology, Tehran, Iran

1 Introduction

Job shop scheduling problem (JSSP), as a well-known branch of scheduling problems, is a combinatorial optimization problem (COP) that deals with allocation of resources to different tasks over time. As a decision-making process, JSSP aims at optimizing one or more objectives. JSSP can be defined as a problem in which n jobs have to be processed by m machines in a prearranged order, and the effort is to find the processing sequences and starting times of each operation on each machine with the aim of optimizing given criteria [3]. The underlying assumptions in JSSPs include the following: (1) Simultaneous processing of different operations in each job is not allowed, (2) each job can visit each machine at most once, and (3) each machine can process at most one job at the same time.

According to Garey et al. [11], JSSPs are NP-complete and among the most difficult COPs. As a result, finding the optimal solutions through mathematical sense is very hard and not always viable in practice. Therefore, heuristic algorithms are commonly used to attain near optimal solutions of JSSPs. Makespan minimization (i.e., minimizing the maximum completion time for processing all the jobs) as the objective of JSSPs has a wide application in academic and industrial practices on account of its simplicity from a mathematical perspective and straightforward formulation [3]. On the other hand, based on the degree of uncertainty associated with input parameters, scheduling problems can be classified into deterministic and nondeterministic categories. The deterministic scheduling models do not take into account uncertainties, while the real-world problems constantly face unstable events such as machine failures, variations in due dates and demands, and fluctuating processing times, which can lead to disruption for working processes [18]. The concepts of fuzzy set theory enable us to model imprecise and uncertain characteristics of real-world situations. Because the uncertainty associated with processing times is quite common in JSSPs, it is more realistic to take them as fuzzy numbers.

This study is a further extension of the work by Azadeh et al. [3]. They optimized the dispatching rule selection through computer simulation and artificial neural networks (ANNs) so as to minimize the makespan in a stochastic environment. Considering processing times as fuzzy numbers to deal with uncertainty, this paper develops a new hybrid algorithm based on computer simulation and adaptive neuro-fuzzy inference system (ANFIS). To the best of our knowledge, this is the first study that presents an approach based on computer simulation and ANFIS for selecting the optimal dispatching rules in an attempt to minimize the makespan in JSSPs.

The remainder of the paper is organized as follows. Section 2 summarizes the relevant literature. Section 3 provides details of the proposed hybrid algorithm. Sections 4 presents the development process of the simulation network for the JSSPs. Computational results are provided in Sect. 5. Last, Sect. 6 is dedicated to the conclusions and directions for future research.

2 Literature review

The literature contains a number of popular papers dealing with JSSPs. For instance, a branch-and-bound algorithm and some heuristic algorithms for JSSPs were proposed by Jurisch [14]. A heuristic algorithm for the design and implementation of a scheduling system with special no-wait or overlapping constraints in a glass factory was developed by Alvarez-Valdes et al. [1]. They attempted to find a schedule with a criterion based on earliness and tardiness penalties. Pan and Huang [22] developed a hybrid genetic algorithm (HGA) to solve a no-wait job shop (NWJS) problem with the aim of minimizing total completion time. Hasan et al. [12] studied JSSPs, aimed to minimize makespan while satisfying several constraints, and developed a memetic algorithm (MA) to solve their problem. Sha and Lin [25] presented a particle swarm optimization (PSO) algorithm for a complex multiobjective JSSP. In order to do this, and on account of the discrete solution spaces of scheduling optimization problems, they had to modify the original PSO (which was used to solve continuous optimization problems). A novel heuristic algorithm based on a filter-and-fan (F&F) procedure was proposed by Rego and Duarte [24] for the JSSP that successfully integrated the classical shifting bottleneck procedure (SBP) with a dynamic and adaptive neighborhood search procedure. Zhou et al. [36] proposed a hybrid structure combining a heuristic and a genetic algorithm (GA) for JSSPs to minimize weighted tardiness. In their paper, they also presented a generalized hybrid framework that was capable of adjusting to a variety of JSSPs with or without sequence-dependent setups and with different objectives (e.g., minimizing makespan, tardiness, flow time, and so forth).

Flexible job shop scheduling problem (FJSP) is an extension of the traditional JSSP. Zhang et al. [35] integrated PSO and tabu search (TS) algorithms to solve the multiobjective FJSP with a number of contradictory and incommensurable objectives. An artificial immune algorithm (AIA) based on integrated approach was developed by Bagheri et al. [4] which uses several policies to produce the initial population and also to select the individuals for reproduction. Xing et al. [32] presented a knowledge-based ant colony optimization (KBACO) algorithm to solve FJSSP. Wang et al. [31] proposed an effective artificial bee colony (ABC) algorithm to deal with FJSP with the objective of minimizing makespan. An FJSP with machine availability constraints was studied by Wang and Yu [30] in which all the machines were prone to preventive maintenance (PM) during the planning period. They presented a filtered beam search (FBS)-based heuristic algorithm to solve this problem.

In the literature, fuzzy sets theory, introduced by Zadeh [33], has been extensively used by many works in the fields of operation research, management science, simulation, and many other fields [2]. For example, Lei [16] investigated the fuzzy JSSP with availability constraints and aimed to find a schedule that maximized the minimum agreement index subject to periodic maintenance, nonresumable jobs, and fuzzy due date. In another paper, Lei [17] presented an FJSP with fuzzy processing times and developed a decomposition-integration genetic algorithm (DIGA) to deal with the problem with the aim of minimizing the maximum fuzzy completion time. Based on simulation experiments, Muhuri and Shukla [20] considered fuzzy processing times and fuzzy due dates with different membership functions to model uncertainty of the real-time tasks. The authors emphasized that based on fuzzy times, there are various satisfaction crossover points of modified task deadlines for distinct membership functions of task deadlines resulting in different satisfactions of scheduling ways of task set. Chen [8] fuzzified the tailored nonlinear fluctuation smoothing rule for mean cycle time (TNFSMCT) for job dispatching in a wafer fabrication factory. Employing fuzzy concepts, numerous new training approaches have been developed, such as ANFIS [13]. ANFIS architecture is composed of both ANN and fuzzy logic involving linguistic expression of membership functions (MFs) and if–then rules. It has been successfully used by many studies (see for example, [5–7]).

Also, simulation has been used in a lot of works on JSSPs. For instance, integrating the simulation and response surface methodology (RSM), Zhang et al. [34] presented a method for assessment and optimization of dispatching rules. Lin et al. [19] developed discrete event simulation models to implement the heuristic dispatching rules in an automated material handling system (AMHS) with a zone control scheme to prevent vehicle collision. In the field of JSSPs, Azadeh et al. [2], for the first time, introduced a flexible intelligent approach for dealing with vagueness and nonlinearity of scheduling problems. They presented a flexible ANN–fuzzy simulation (FANN–FS) algorithm to solve this multiattribute combinatorial dispatching (MACD) decision problem.

Several recent studies have employed the combination of neural networks and fuzzy rules as in ANFIS for JSSPs. According to Dong and Liu [9], this is due to some practical characteristics of ANFIS such as self-learning, nonlinear mapping, and the form of if–then fuzzy rules. Dong et al. [10] utilized ANFIS to combine the heuristic rules and obtained some fuzzy rules in JSSPs with parallel machines. Using simulation, they showed that the ANFIS-based adaptive algorithm combines the rules properly and outperforms heuristics rules. Dong and Liu [9] employed ANFIS to combine the heuristic rules nonlinearly and took it as an adaptive scheduling rule to sort jobs. In the work by Shafaei et al. [26], six heuristic algorithms along with ANFIS were used for estimating the makespan to solve a no-wait two-stage flexible flow shop. Table 1 summarizes the related literature.

In this paper, the inherent uncertain nature of input parameters is represented as fuzzy processing times. The proposed algorithm uses the combination of neural networks and fuzzy rules in ANFIS approach together with discrete event simulation and attempts to minimize makespan by selecting optimum dispatching rules in JSSPs. For this purpose, simulation is first developed to evaluate different permutations of dispatching rules. Subsequently, based on the results obtained, a feed forward ANFIS is employed to search the solution space and achieve the global optimal solution.

3 The proposed hybrid algorithm

In this section, a new hybrid algorithm based on discrete event simulation and ANFIS is presented to find the optimal solution to JSSPs in an uncertain environment. The priority rules are considered nonidentical, and the variability in the processing times is taken into account in the form of fuzzy numbers. This algorithm aims to select the best set of dispatching rules among a set of assigned rules to each machine in an attempt to minimize the makespan. The following steps describe the general framework of the proposed algorithm:

Step 1 Build the discrete event simulation network of the JSSP using the problem's data including parameters of fuzzy processing times on each machine, numbers of the machines and jobs, and machining sequence of the jobs.

Step 2 Define the following ten priority rules to allocate to each machine [27]. Since this is an actual system, the experts of the system have identified ten dispatching rules as follows. In these rules, "TNOW" denotes the current time of simulation. "Number of machines" shows the number of all machines involved in the

Table 1 Overview of the related literature

Study	Objective	Methodology	Data complexity and nonlinearity	Fuzzy processing times	Global optimization	Intelligent modeling and forecasting
Zhang et al. [34]	Evaluation and optimization of dispatching rules	Simulation and response surface methodology (RSM)	✓		✓	
Azadeh et al. [3]	Minimizing makespan	Simulation and ANNs	✓		✓	✓
Orides et al. [21]	AGVs' assignment	Genetic algorithm (GA)			✓	
Sha and Lin [25]	Multi-objective optimization	Particle swarm optimization (PSO)				
Shafaei et al. [26]	Minimizing makespan	Six heuristic algorithms-ANFIS	✓	✓	✓	
Wang et al. [31]	Minimizing the maximum makespan	Artificial bee colony (ABC)				
Lei [16]	Minimizing the maximum fuzzy completion time	GA	✓	✓		
Hasan et al. [12]	Minimizing makespan	Memetic algorithm (MA)				
This study	Optimizing dispatching rules with the aim of minimizing makespan	Computer Simulation- ANFIS	✓	✓	✓	✓

processing for each job. "Jobstep" represents the current step of processing for each job. Thus, the expression "number of machines-jobstep" indicates the remaining steps for each job. The term "proctime" represents the mean of the processing times of the jobs. These rules are applied where one machine is available and more than one job can be processed on it to achieve a better solution.

(1) FIFO (first in first out);
(2) LIFO (last in first out);
(3) LVF (low value first) on (proctime);
(4) HVF (high value first) on (proctime);
(5) HVF on $\left(\frac{\text{number of machines} - \text{jobstep}}{\text{proctime}}\right)$
(6) LVF on $\left(\frac{\text{number of machines} - \text{jobstep}}{\text{proctime}}\right)$;
(7) HVF on $(\text{number of machines} - \text{jobstep})$;
(8) LVF on $(\text{number of machines} - \text{jobstep})$;
(9) HVF on $\left(\frac{\text{number of machines} - \text{jobstep}}{\text{TNOW} + \text{proctime}}\right)$;
(10) LVF on $\left(\frac{\text{number of machines} - \text{jobstep}}{\text{TNOW} + \text{proctime}}\right)$.

Step 3 Run the simulation model to obtain the makespan value for an adequate number of different sets of permutation of priority rules (here, 300 sets of permutations are considered). For this purpose, allocate values between 1 and 10 to each machine, arbitrarily. For example, a problem with five machines and the assigned set [2 5 3 1 4] implies that a solution with priority rules 2, 5, 3, 1, and 4 has been allocated to machines 1 to 5, respectively.

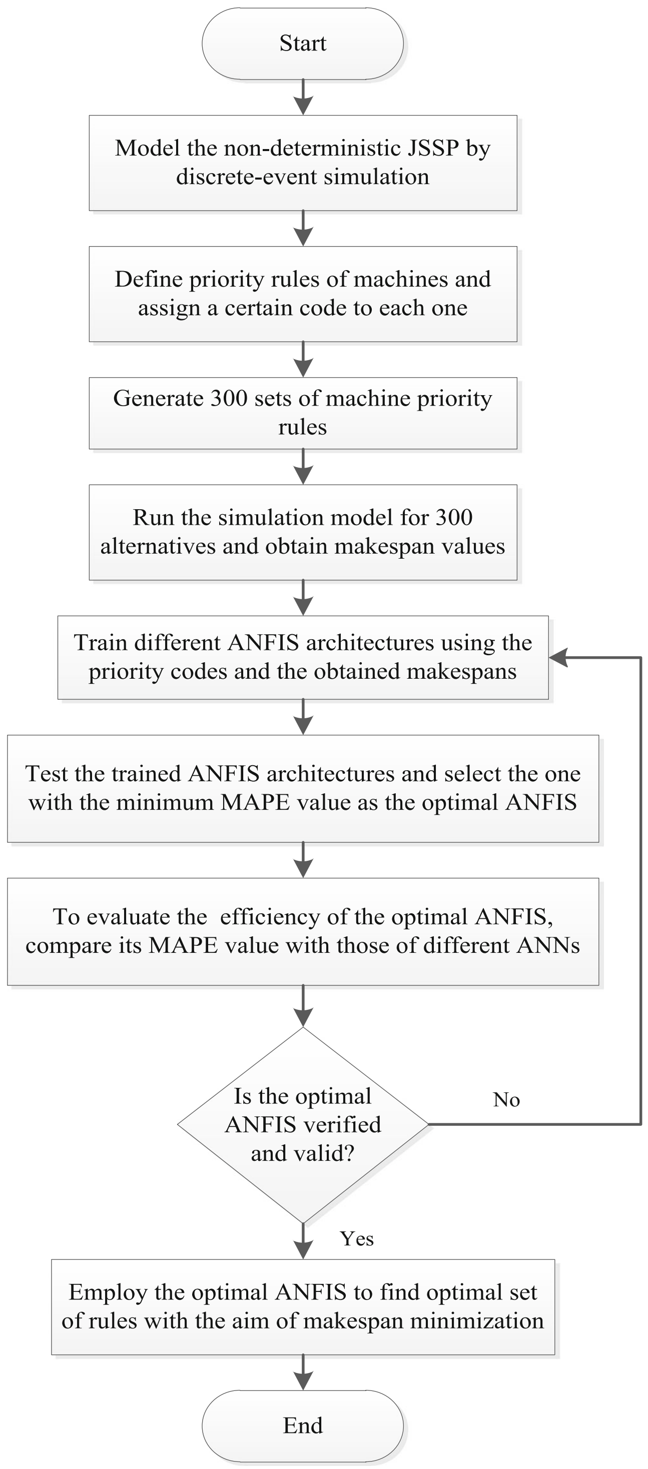
Step 4 Assign 85 % of the output data set yielded by the simulation model (i.e., makespan) together with the relevant input data as the training data to train different architectures of ANFIS.

Step 5 Apply the remaining 15 % of the data as the test data to test the trained ANFIS models, and calculate their corresponding mean absolute percentage error (MAPE) values as the relative error. Thereafter, select the ANFIS model with the minimum MAPE value and call it optimal ANFIS.

Step 6 In order to examine the efficiency of the optimal ANFIS, design several multilayer perceptron (MLP) ANNs and perform the same procedure on them, and compare their MAPE results with that of the optimal ANFIS.

Step 7 Having trained and tested the ANFIS, run the optimal ANFIS to estimate the makespan values related to all possible permutations of priority rules and specify the minimum one.

Generally speaking, it is impossible to use simulation approach solely to evaluate all solutions of a JSSP even with a very few number of machines. Because, by considering nonidentical machine priority rules for a problem with n machines, there are 10^n solutions, resulting in, for example, 10,000 times running the simulation model for a problem with four machines. ANFIS, as a fuzzy inference network, removes the need of simulating all permutations of priority rules. The structure of the proposed algorithm is depicted in Fig. 1.


Start
Model the non-deterministic JSSP by discrete-event simulation
Define priority rules of machines and assign a certain code to each one
Generate 300 sets of machine priority rules
Run the simulation model for 300 alternatives and obtain makespan values
Train different ANFIS architectures using the priority codes and the obtained makespans
Test the trained ANFIS architectures and select the one with the minimum MAPE value as the optimal ANFIS
To evaluate the efficiency of the optimal ANFIS, compare its MAPE value with those of different ANNs
Is the optimal ANFIS verified and valid?
No
Yes
Employ the optimal ANFIS to find optimal set of rules with the aim of makespan minimization
End
Fig. 1 Diagram of the proposed hybrid algorithm

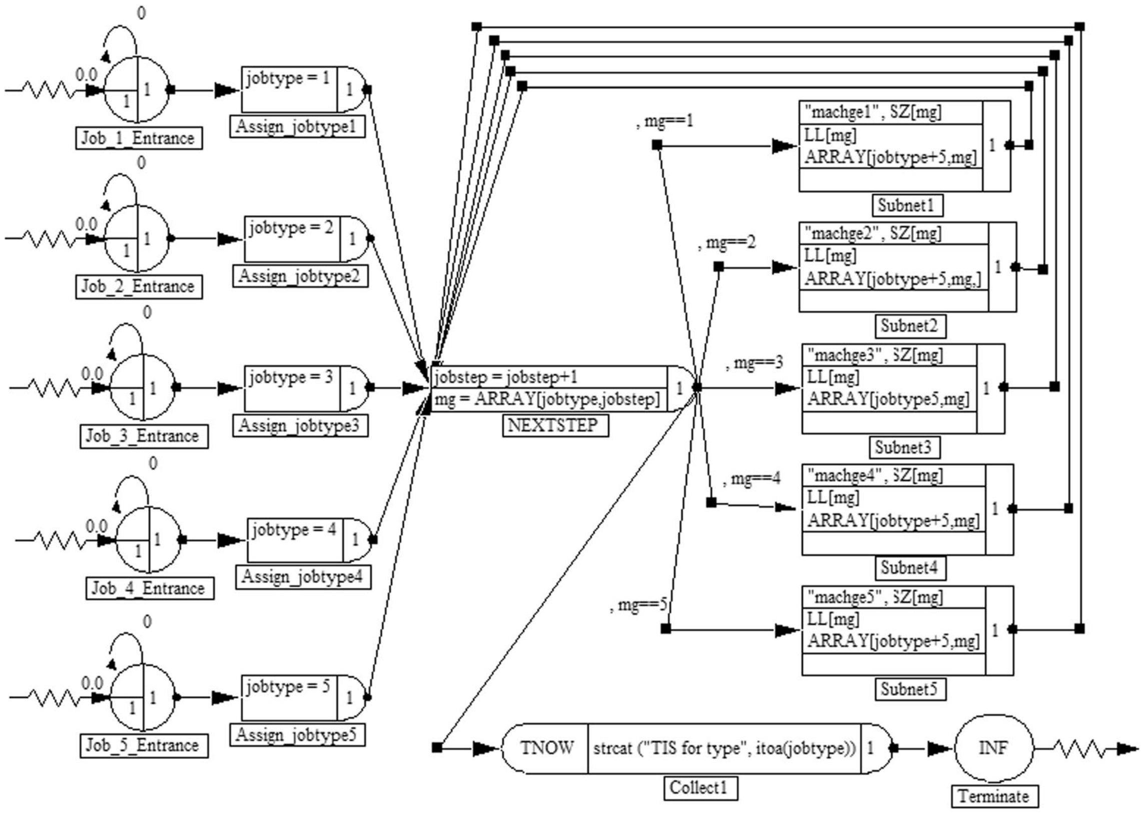


Fig. 2 Simulation network designed by Visual SLAM® for a JSSP with five jobs and five machines

4 Simulation network modeling

Simulation is a powerful tool for understanding the behavior of systems and analyzing scenarios [23]. It has extensively been applied by previous studies for solving scheduling problems (see for example, [15,28,29]). Figure 2 depicts a network-based modeling for a fuzzy JSSP with five jobs and five machines by Visual Slam simulation software. In this model, jobs and machines are taken as entities and servers, respectively. Each entity with a specific attribute representing its type is emanated in the network by a CREATE node. Located after the CREATE node is the ASSIGN node which determines the type of jobs. A full description of all the nodes in Fig. 2 is provided in Pritsker and O'Reilly [23]. After freeing a machine based on the predefined priority rule, a job is selected and allocated to it and the other jobs wait in the file number of machines. AWAIT node, located in the corresponding subnetwork, performs this process. In this study, there are five subnetworks, each modeling a machine. This approach allows assigning a priority rule to each machine [3]. The subnetwork related to machine 1 is illustrated in Fig. 3. As mentioned earlier, processing times are taken as fuzzy numbers. The lower and higher bounds for processing times of each machine are defined in accordance with each job to represent variability. A proctime variable sorts the mean of each processing time on each machine.

Using the FREE node, the operation is declared to be completed, and the machine is available to process the next operation. This procedure is replicated until all operations are processed by machines. The time of system for all jobs is

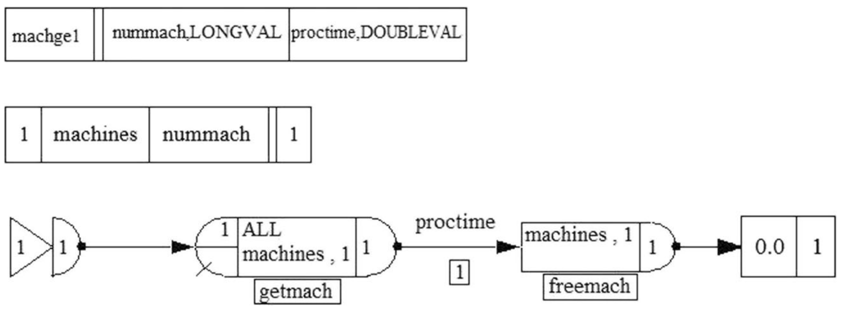


Fig. 3 Visual SLAM® subnetwork model for machine 1

Table 2 Visual SLAM® original network module definitions

Label	Node	Definition
Job_1_Entrance	CREATE	Job-type 1 arrives
Job_2_Entrance	CREATE	Job-type 2 arrives
Job_3_Entrance	CREATE	Job-type 3 arrives
Job_4_Entrance	CREATE	Job-type 4 arrives
Job_5_Entrance	CREATE	Job-type 5 arrives
Assign_jobtype1	ASSIGN	The label of "type 1" is assigned to the job
Assign_jobtype2	ASSIGN	The label of "type 2" is assigned to the job
Assign_jobtype3	ASSIGN	The label of "type 3" is assigned to the job
Assign_jobtype4	ASSIGN	The label of "type 4" is assigned to the job
Assign_jobtype5	ASSIGN	The label of "type 5" is assigned to the job
NEXTSTEP	ASSIGN	Job types are assigned to the machines
Subnet1	CALLVSN	Calls VSN 1 and sends entities (jobs) to it
Subnet2	CALLVSN	Calls VSN 2 and sends entities (jobs) to it
Subnet3	CALLVSN	Calls VSN 3 and sends entities (jobs) to it
Subnet4	CALLVSN	Calls VSN 4 and sends entities (jobs) to it
Subnet5	CALLVSN	Calls VSN 5 and sends entities (jobs) to it
No label	VSN	Defines VSN 1 name and its parameters
No label	VSN	Defines VSN 2 name and its parameters
No label	VSN	Defines VSN 3 name and its parameters
No label	VSN	Defines VSN 4 name and its parameters
No label	VSN	Defines VSN 5 name and its parameters
No label	RESOURCE	The resource "machine 1" and its capacity
No label	RESOURCE	The resource "machine 2" and its capacity
No label	RESOURCE	The resource "machine 3" and its capacity
No label	RESOURCE	The resource "machine 4" and its capacity
No label	RESOURCE	The resource "machine 5" and its capacity
Enter VSN	ENTERVSN	Enters elected jobs to machine 1 sub-network
Enter VSN	ENTERVSN	Enters elected jobs to machine 2 sub-network
Enter VSN	ENTERVSN	Enters elected jobs to machine 3 sub-network
Enter VSN	ENTERVSN	Enters elected jobs to machine 4 sub-network
Enter VSN	ENTERVSN	Enters elected jobs to machine 5 sub-network
getmach	AWAIT	Asks for machine 1 to be processed
getmach	AWAIT	Asks for machine 2 to be processed
getmach	AWAIT	Asks for machine 3 to be processed
getmach	AWAIT	Asks for machine 4 to be processed
getmach	AWAIT	Asks for machine 5 to be processed
freemach	FREE	Frees machine 1
freemach	FREE	Frees machine 2
freemach	FREE	Frees machine 3
freemach	FREE	Frees machine 4
freemach	FREE	Frees machine 5
RTRNVSN	RETURNVSN	Determines return and exit rules for VSN 1
RTRNVSN	RETURNVSN	Determines return and exit rules for VSN 2
RTRNVSN	RETURNVSN	Determines return and exit rules for VSN 3
RTRNVSN	RETURNVSN	Determines return and exit rules for VSN 4
RTRNVSN	RETURNVSN	Determines return and exit rules for VSN 5
Collect1	COLCT	Collects time in system for all job types
Terminate	TERMINATE	Destroys entity and completes simulation

accumulated via a COLCT node, and its report is printed to an output file. As all jobs enter the TERMINATE node, the simulation will be completed. Table 2 gives the label and role of all nodes in the Visual SLAM. For more information regarding modeling and simulation by Visual SLAM, the interested readers are referred to Pritsker and O'Reilly [23].

5 Experimental results

This section provides the results of applying the proposed approach on a 5×5 testbed problem. As mentioned earlier, processing times are considered in the form of triangular fuzzy numbers. The lower and higher bounds for processing times of each machine are defined with respect to each job. They specify the variability of the processing times. Table 3 shows the processing times of machines on the jobs. Also, the machine sequence for each job is provided in Table 4.

The problem data are used to develop the simulation model using an appropriate control statement. The control statement contains information regarding machine sequences and processing times. INTLC statement allocates values to string variables such as the resource names. It is favorable that time of the system (i.e., makespan) be minimized by achieving the optimal set of priority rules. Because processing times are fuzzy numbers changing between their lower and higher bounds, the simulation outputs, e.g., makespan, are not constant values in all the replications of the simulation model. Hence, to ensure the robustness of the simulation results, the model is replicated 1000 times for each possible set of dispatching rules, and the average of outputs of 1000 runs is considered as the final makespan.

The simulation results (i.e., makespan values) of testbed problem in visual SLAM obtained for 300 different permutations of machine priority rules are used for training and testing different ANFIS architectures. To this end, after implementing the simulation model, the input data in conjunction with the corresponding outputs of the simulation model are used to develop the ANFIS model. Note that 187 data sets (i.e., 85 % of the total) are selected for training the ANFIS architectures, and the rest (i.e., 33 data sets) are used to test them. Moreover, we have five jobs and five machines, and each job has a processing time on each machine. Thus, there are 25 inputs related to processing times and 5 inputs for dispatching rules for ANFIS resulting in 30 inputs. Makespan values yielded by running the simulation are the only outputs of the ANFIS. It should be mentioned that the processing times of the jobs on machines are divided into nine values between the relevant ranges. For example, the divisions of the processing time of job 5 on machine 1 are 6, 6.25, 6.5, 6.75, 7, 7.25, 7.5, 7.75, and 8. These values are arbitrarily selected as the inputs for ANFIS models.

Different ANFIS architectures are designed, and the value of MAPE related to test data is calculated for each of them. It should be mentioned that, in order to handle possible noise, each ANFIS architecture is run 50 times and the average MAPE value is considered as the final MAPE. The ANFIS model with the minimum MAPE value is selected as the optimal ANFIS. In order to validate the optimal ANFIS model, we perform the same procedure and obtain MAPE values for different MLP-ANN structures and compare the results with those of the optimal ANFIS. MATLAB software package is applied for codification. Tables 5 and 6 present the respective results for different ANFIS structures and ANNs. Note that the very 187 data sets are used for training and validating ANNs and the remaining 33 data sets are meant for testing. As is clear

Table 3 Processing times of machines for jobs 1–5

Job	Machine	Lower bound	Mean	Higher bound
1	1	5	7	9
	2	1	2	3
	3	8	10	12
	4	3	4	5
	5	2	3	4
2	1	4	5	6
	2	0.5	1	1.5
	3	6	8	10
	4	4	6	8
	5	8	10	12
3	1	1	2	3
	2	6	8	10
	3	5	6	7
	4	2	3	4
	5	2	3	4
4	1	3	4	5
	2	6	8	10
	3	6	7	8
	4	1	2	3
	5	8	9	10
5	1	6	7	8
	2	9	10	11
	3	4	5	6
	4	4	5	6
	5	6	8	10

Table 4 Machine sequences

Jobs	Machine sequence
1	3–2–4–1–5
2	3–5–1–4–2
3	5–2–1-3–4
4	1–3–4–5–2
5	2–5–4–3–1

Table 5 Architecture of different ANFIS architectures and their associated relative error (MAPE)

Input MF	Output MF	Initial FIS	N of Cls	N of MFs	R	Opt.	And	Or	Imp.	Agg.	MAPE
G	L	SC			0.7	H	Min	Max	Min	Max	0.161
G	L	SC			0.6	BP	Min	Max	Prod	Sum	0.153
G	L	SC			0.3	BP	Min	Probor	Min	Sum	0.211
G	L	SC			0.4	H	Prod	Max	Min	Sum	0.206
G	L	SC			0.5	BP	Prod	Max	Prod	Sum	0.172
G	L	SC			0.2	H	Prod	Probor	Min	Max	0.164
Tri	C	GP		2		H	Prod	Max	Min	Sum	0.267
G	L	GP		3		BP	Min	Max	Prod	Max	0.148
G	C	GP		2		H	Prod	Max	Prod	Sum	0.161
Tri	L	GP		3		BP	Min	Probor	prod	Max	0.224
Tri	C	GP		2		H	Min	Probor	Prod	Sum	0.182
G	L	GP		3		BP	Prod	Probor	Min	Max	0.177
G	L	FC	7			BP	Min	Max	Min	Max	0.152
G	L	FC	5			H	Prod	Max	Min	Sum	0.155
G	L	FC	6			BP	Min	Probor	Min	Max	**0.128**
G	L	FC	8			BP	Min	Max	Prod	Max	0.166
G	L	FC	4			H	Prod	Probor	Prod	Sum	0.148
G	L	FC	9			H	Prod	Probor	Min	Sum	0.139

SC subtractive clustering, *GP* grid partition, *FC* fuzzy c-means (FCM) clustering, *H* hybrid, *BP* backpropagation, *R* radius, *FIS* fuzzy inference system, *MF* membership function, *Cl* cluster, *Opt* optimization method, *Imp.* implication method, *Agg.* aggregation method, *G* gaussmf, *Tri* trimf, *L* linear, *C* constant, *N* number

The bold entry shows the best ANFIS architecture

Table 6 Architectures of different MLP-ANN models and their associated relative error (MAPE)

MLP ANN No.	Training function	Transfer function of the hidden layer	Number of neurons in the hidden layer	Transfer function of the output layer	MAPE
1	LM	logsig	8	purelin	0.151
2	GDA	tansig	5	purelin	0.171
3	BFGS	tansig	7	purelin	0.155
4	OSS	logsig	15	purelin	0.173
5	GD	tansig	12	purelin	0.199
6	LM	logsig	4	purelin	0.165
7	GDX	tansig	9	purelin	0.180
8	GDA	tansig	12	purelin	0.181
9	BFGS	logsig	6	purelin	0.159
10	OSS	logsig	14	purelin	0.190
11	LM	tansig	11	purelin	0.172
12	GDX	logsig	9	purelin	0.187

LM Levenberg–Marquardt backpropagation, *GD* gradient descent backpropagation, *OSS* one-step secant backpropagation, *GDA* gradient descent with adaptive learning rule backpropagation, *BFGS* quasi-Newton backpropagation, *GDX* gradient descent with momentum and adaptive learning rule back propagation

from these tables, the optimal ANFIS possesses a smaller MAPE value (i.e., 0.128) compared to all ANN structures provided in Table 6, and thus is verified. Moreover, Fig. 4 depicts the graphical representation of comparison between the actual makespan values obtained by simulation and those predicted by optimal ANFIS. The visual fitness between the actual and predicted values as well as the MAPE values indicate that the simulation model was capable of providing a suitable training set to develop reliable ANFIS, resulting in efficient estimation of the makespan values for all possible situations (i.e., all permutations of priority rules).

In this step, optimal ANFIS is prepared to be applied to determine the estimated makespan values for all permutations of machine priority rules. Table 7 gives some of the resultant makespan values. Using the proposed hybrid algorithm, the local optimal makespan value is equal to 48 min for the 5×5 problem, while the near optimal makespan values obtained by computer simulation method [27] and simulation-ANN [3] were 50.438 and 48.803 min, respectively.

Practically speaking, using computer simulation solely, numerous outputs (for example 100,000 solutions for a problem with five machines) are required to find in order to achieve the optimal solution. This is a very time-consuming process. The

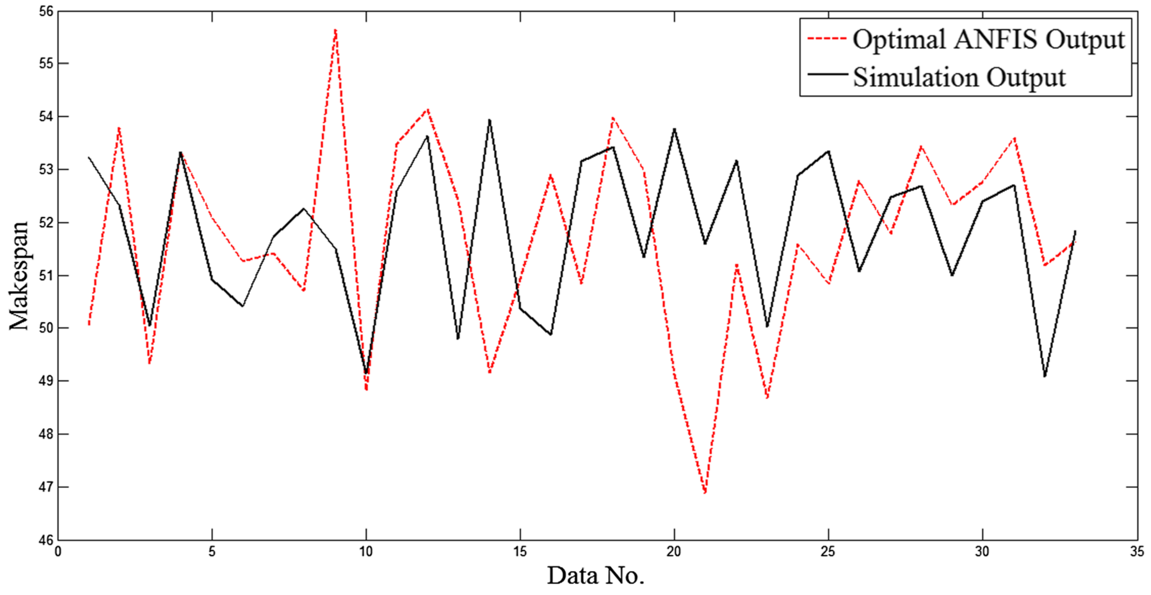


Fig. 4 Optimal ANFIS output versus the simulation output (actual value)

proposed hybrid algorithm significantly reduces the time required to select the optimal set of dispatching rules and improves attaining optimal solutions for small-sized JSSPs under uncertainty. The CPU time required for training the network and searching the solution space was 1280.44 s for the 5×5 fuzzy JSSP. Thus, the proposed hybrid algorithm improves the process of selecting optimal dispatching rules in terms of computational time and achieving optimal solutions for small-sized JSSPs under uncertainty.

6 Conclusions and directions for future research

This paper dealt with minimizing makespan for JSSPs in an uncertain environment. Unstable conditions in JSSPs such as machine failure, operator unavailability, and various due dates lead to accompanied uncertainty with input parameters. These characteristics add to variability, complexity, and nonlinearity of the problem. A new hybrid algorithm based on computer simulation and ANFIS was proposed in this study to consider

Table 7 All solutions of the 5×5 fuzzy JSSP estimated by the optimal ANFIS

Permutation no.	Dispatching rule					Makespan
	machine 1	machine 2	machine 3	machine 4	machine 5	
1	1	8	3	5	6	63
2	2	3	6	1	3	51
3	9	2	4	7	9	61
4	8	7	5	6	1	62
5	9	9	8	3	3	48
6	9	2	4	7	9	53.5
7	4	2	1	9	2	66.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮
99993	6	2	10	1	8	48.25
99994	10	10	10	9	5	48
99995	2	7	7	7	7	55.5
99996	4	10	9	7	3	55.5
99997	5	2	1	7	4	63.625
99998	4	10	9	7	3	48
99999	4	2	4	8	9	46.5
100000	4	2	6	2	1	68

several dispatching rules in JSSPs, simultaneously, and to minimize makespan under uncertain processing times. Processing times were assumed to be fuzzy numbers. A JSSP with five machines and five jobs was considered as our test problem. Discrete event simulation was employed due to its capability to handle complex and nonlinear problems. Three hundred sets of priority rules in conjunction with their corresponding makespan values obtained by the simulation model were taken as the input data for training and testing different architectures of ANFIS. The ANFIS structure with the minimum relative error (MAPE) was considered as the optimal ANFIS. Thereafter, in order to test the efficiency of the optimal ANFIS, it was compared to different structures of multilayer perceptron ANNs in terms of MAPE value. The optimal ANFIS was then employed to search the solution space and estimate the makespan values of all possible sets of dispatching rules to achieve the local optimal solution of the JSSP.

Based on the results obtained, the proposed algorithm proved to be an efficient structure for minimizing the makespan in JSSPs under uncertainty. It attempts to reduce the computational time considerably and find the optimal solution for small-sized JSSPs under uncertainty. The hybrid computer simulation-ANFIS algorithm provides higher precision via preprocessing (training) and postprocessing (testing) the given data. It is also capable of capturing the uncertainty, complexity, and nonlinearity of the environment. This algorithm helps managers to assess the performance of the job shop systems by assigning various priority rules to machines. Although it practically finds the optimal solution within a reasonable period of time for small-sized problems, it cannot be properly applied to large-scale problems with numerous machines and great solution spaces. It should be mentioned that, because of the ability of ANFIS to handle multiple outputs, the method can consider several objective functions. Therefore, the study of bicriteria and multicriteria JSSPs under uncertainty could be considered as further research.

Acknowledgments The authors are grateful for the valuable comments and suggestion from the respected reviewers. Their valuable comments and suggestions have enhanced the strength and significance of our paper. This study was supported by a grant from University of Tehran (Grant No. 8106013/1/19). The authors are grateful for the support provided by the College of Engineering, University of Tehran, Iran.

References

1. Alvarez-Valdes R, Fuertes A, Tamarit JM, Giménez G, Ramos R (2005) A heuristic to schedule flexible job-shop in glass factory. Eur J Oper Res 165:525–534
2. Azadeh A, Moghaddam M, Geranmayeh P, Naghavi A (2010) A flexible artificial neural network–fuzzy simulation algorithm for scheduling a flow shop with multiple processors. Int J Adv Manuf Technol 50:699–715
3. Azadeh A, Negahban A, Moghaddam M (2012) A hybrid computer simulation-artificial neural network algorithm for optimisation of dispatching rule selection in stochastic job shop scheduling problems. Int J Prod Res 50(2):551–566
4. Bagheri A, Zandieh M, Mahdavi I, Yazdani M (2010) An artificial immune algorithm for the flexible job-shop scheduling problem. Futur Gener Comput Syst 26(4):533–541
5. Cakmakci M (2007) Adaptive neuro-fuzzy modelling of anaerobic digestion of primary sedimentation sludge:Bioprocess and. Biosyst Eng 30(5):349–357
6. Chang FJ, Chang YT (2006) Adaptive neuro fuzzy inference system for prediction of water level in reservoir. Adv Water Resour 29(1):1–10
7. Chau KW, Wu CL, Li YS (2005) Comparison of several flood forecasting models in Yangtze River. J Hydrol Eng 10(6):485–491
8. Chen T (2012) A fuzzy fluctuation smoothing rule for job dispatching in a wafer fabrication factory: a simulation study. Int J Fuzzy Syst Appl (IJFSA) 2(4):47–63
9. Dong M, Liu M (2011) An ANFIS based dispatching rule for complex fuzzy job shop scheduling problem: Proceedings of the IEEE International Conference on Information Science and Technology (ICIST) 263–266
10. Dong M, Liu M, Wu C (2005) An ANFIS based adaptive algorithm for job shop scheduling problem with parallel machines: control engineering of China
11. Garey M, Johnson D, Sethi R (1976) The complexity of flow shop and job shop scheduling. Math Oper Res 1(2):117–129
12. Hasan SK, Sarker R, Essam D, Cornforth D (2009) Memetic algorithms for solving job-shop scheduling problems. Memet Comput 1(1):69–83
13. Jang JS (1993) ANFIS: adaptive-network-based fuzzy inference system:Systems, Man and Cybernetics. IEEE Trans 23(3):665–685
14. Jurisch B (1992) Scheduling jobs in shops with multi-purpose machines, PhD thesis. University of Osnabrük, Germany
15. Kadipasaoglu SN, Xiang W, Khumawala BM (1997) A comparison of sequencing rules in static and dynamic hybrid flow systems. Int J Prod Res 35(5):1359–1384
16. Lei D (2010) Fuzzy job shop scheduling problem with availability constraints. Comput Ind Eng 58(4):610–617
17. Lei D (2010) A genetic algorithm for flexible job shop scheduling with fuzzy processing time. Int J Prod Res 48(10):2995–3013
18. Lejmi T, Sabuncuoglu I (2002) Effect of load, processing time and due date variation on the effectiveness of scheduling rules. Int J Prod Res 40(4):945–974
19. Lin JT, Wang FK, Yen PY (2001) Simulation analysis of dispatching rules for an automated interbay material handling system in wafer fab. Int J Prod Res 39(6):1221–1238
20. Muhuri PK, Shukla KK (2009) Real-time scheduling of periodic tasks with processing times and deadlines as parametric fuzzy numbers. Appl Soft Comput 9:936–946
21. Orides M, Castro PAD, Kato ER, Camargo HA (2006) A genetic fuzzy system for defining a reactive dispatching rule for AGVs: In Systems, Man and Cybernetics, 2006. SMC'06. IEEE Int Conf 1:56–61
22. Pan JC-H, Huang H-C (2009) A hybrid genetic algorithm for no-wait job shop scheduling problems. Expert Syst Appl 36(3):5800–5806
23. Pritsker AAB, O'Reilly JJ (1999) Simulation with Visual SLAM® and AweSim®. John Wiley and Sons, New York
24. Rego C, Duarte R (2009) A filter-and-fan approach to the job shop scheduling problem. Eur J Oper Res 194(3):650–662
25. Sha D, Lin H-H (2010) A multi-objective PSO for job-shop scheduling problems. Expert Syst Appl 37(2):1065–1070
26. Shafaei R, Rabiee M, Mirzaeyan M (2011) An adaptive neuro fuzzy inference system for makespan estimation in multiprocessor no-wait two stage flow shop. Int J Comput Integr Manuf 24(10):888–899

27. Tavakkoli-Moghaddam R, Daneshmand-Mehr M (2005) A computer simulation model for job shop scheduling problems minimizing makespan. Comput Ind Eng 48(4):811–823
28. Vinod V, Sridharan R (2008) Dynamic job-shop scheduling with sequence-dependent setup times: simulation modeling and analysis. Int J Adv Manuf Technol 36(3–4):355–372
29. Vinod V, Sridharan R (2009) Simulation-based meta-models for scheduling a dynamic job shop with sequence-dependent setup times. Int J Prod Res 47(6):1425–1447
30. Wang S, Yu J (2010) An effective heuristic for flexible job-shop scheduling problem with maintenance activities. Comput Ind Eng 59(3):436–447
31. 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(1–4):303–315
32. Xing L-N, Chen Y-W, Wang P, Zhao Q-S, Xiong J (2010) A knowledge-based ant colony optimization for flexible job shop scheduling problems. Appl Soft Comput 10(3):888–896
33. Zadeh LA (1965) Fuzzy sets. Inf Control 8(3):338–353
34. Zhang H, Jiang Z, Guo C (2009) Simulation-based optimization of dispatching rules for semiconductor wafer fabrication system scheduling by the response surface methodology. Int J Adv Manuf Technol 41(1–2):110–121
35. Zhang G, Shao X, Li P, Gao L (2009) An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. Comput Ind Eng 56(4): 1309–1318
36. Zhou H, Cheung W, Leung LC (2009) Minimizing weighted tardiness of job-shop scheduling using a hybrid genetic algorithm. Eur J Oper Res 194(3):637–649