Journal of Ambient Intelligence and Humanized Computing (2023) 14:2657–2671
https://doi.org/10.1007/s12652-023-04511-1

ORIGINAL RESEARCH

Multi-objective collaborative job shop scheduling in a dynamic environment: Non-dominated sorting memetic algorithm

N. Bagheri Rad[1] · **J. Behnamian**[1]

Received: 17 February 2022 / Accepted: 2 January 2023 / Published online: 7 January 2023

Abstract
In this paper, a novel mathematical model is developed for the distributed multi-agent network scheduling problem in a dynamic job shop environment with availability constraints and new job arrivals. In a dynamic collaborative-competitive environment, a number of factories with independent ownership are merged to form a multi-agent production network in which each production agent, despite participation, seeks to optimize its own objective function as a primary priority. To minimize the makespan and the total energy consumption, the ε-constraint approach is used. Since the problem is NP-hard, a memetic algorithm based on a non-dominated sorting genetic algorithm-II and local search are proposed. Finally, the proposed algorithm is compared with a hybrid Pareto-based tabu search algorithm (HPTSA). The obtained results show that in large-size instances, our proposed algorithm outperforms the HPTSA.

Keywords Distributed scheduling · Job shop · Multi-agent system · Memetic algorithm · Dynamic environment · Multi-objective optimization

1 Introduction

Today, production scheduling in a multi-factory production environment is expressed as a distributed scheduling problem. By developing a culture of cooperation in the production of different products in several factories, distributed scheduling is very useful for better use of resources, increased productivity and profit, reduction of production costs, reduction of risks, and increased product quality. It can be said that a multi-factory production environment has become more important. In recent years, production in a distributed environment in a multi-factory production network has been one of the most attractive issues (Lohmer and Lasch 2021). Also, due to the globalization process in the field of production, the job shop production system is one of the most important and practical topics that has been studied in the distributed scheduling problem. Therefore, due to the practicality of this environment, in this research, a job shop production system is assumed. Applications of the distributed job shop scheduling problem include auto parts manufacturing, home appliances, etc. Sharing distributed resources in different geographical locations results in production networks. Two types of distribution networks have been introduced for multi-factory environments: (i) All factories belong to a single organization. They collaborate with each other to maximize the organization's profits. In this type of collaboration, despite the goal of maximizing profits, some factories may be unprofitable. (ii) A number of factories with independent ownerships make up a multi-factory production network which is called a virtual production network. Each factory in this type of network focuses on its own interests. In this research, it is assumed that the distribution network in the multi-factory environment is of the second type. In this type, the factories can achieve more benefits than the separate mode. In this problem, each factory is considered as an agent. In multi-agent scheduling, each agent has a set of jobs and wants to independently optimize its own objective function. Applications of this scheduling include the rail transport industry, aerial industry, and project-based organizations (Agnetis et al. 2014).

With the rapid development of globalization and information technology, quick response to the diverse needs of customers has become an important issue for modern manufacturing companies (Goli et al. 2019, 2021). In the real world, factories face various unpredictable disruptions

✉ J. Behnamian
Behnamian@basu.ac.ir

[1] Department of Industrial Engineering, Faculty of Engineering, Bu-Ali Sina University, Hamedan, Iran

in the production line, such as machine breakdowns, the arrival of new jobs, changes in delivery times, changes in the number of available employees, etc. In this paper, it is assumed that the arrival of new jobs and the unavailability of machines constraints at some time may occur in the production process. These events can disrupt the production system. Therefore, in order to prevent these changes, it is usually necessary to reschedule after the initial scheduling. Obviously, considering non-availability constraints reduces the sustainability of scheduling. The availability of machine constraints in manufacturing systems is classified into two categories, fixed and non-fixed. In fixed availability, the non-availability period starts and ends at fixed points in time. But in the non-fixed availability, the start time of the period is non-fixed. Furthermore, there are three policies in the face of periods of non-availability of machines due to breakdowns and repair: resumable, semi-resumption and no resumption (Lee 1999). Resumable means that job processing continues without wasting time after the end of the non-availability of machines period (Wang and Yu 2010). In semi-resumption after the end of the inaccessibility period, a percentage of the processing that was done before the inaccessibility period should be done in part (Lee 1999). In the resumption strategy, after the machine is available, the job is processed from the beginning (Allaoui and Artiba 2006). In this paper, it is assumed that the availability constraint is of non-fixed type and the policy is resumable.

According to the above, the main purpose of the research is to define, model and solve the distributed multi-agent network scheduling problem considering the real events such as machine breakdowns and the arrival of new jobs in the job shop production environment. The main research questions are (i) How to model and solve complex scheduling problems in distributed multi-agent systems? (ii) Considering the real events such as machine breakdowns and the arrival of new jobs, what changes and effects will they have on the job shop scheduling? (iii) Is the proposed approach, an efficient approach? Due to the novelty of the problem and, as a result, not being present in the literature, a mixed-integer mathematical model was developed for the problem. Also, due to the complex nature of the problem, a non-dominated sorting memetic algorithm has been proposed to solve it. According to the above explanations, research innovations are (i) considering the multi-agent approach with the virtual alliance in the distributed job shop scheduling problem, (ii) Considering the constraints of machine availability and arrival of new jobs, (iii) Development of a mathematical model, and (iv) Proposing a memetic algorithm based on a non-dominated sorting genetic algorithm-II (NSGA-II) and local search.

The rest of this paper is organized as follows. Section 2 presents a literature review of the problem. Section 3 introduces distributed multi-agent network scheduling problem and proposes mathematical modeling. Section 4 proposes the memetic algorithm (MA) and its structure. Section 5 presents the computational experiments. Finally, Sect. 6 provides the conclusion of the work and future research.

2 Literature review

In recent years, the multi-factory production system has attracted the attention of many researchers and manufacturers. This production system enables factories to respond faster to rapid environmental changes and maintain and increase their competitiveness in the global markets. In this system, factories must work in harmony, to be able to ensure a secure flow of goods, services, and information. In many of these studies in this field, the production environment is considered non-distributed. Williams was as first study in 1981, considered the multi-factory scheduling problem (Williams 1981). In this research, joint scheduling in production and distribution problem in complex networks with the objective of minimizing production and distribution costs in each time period has been investigated. Single machine, parallel, job shop and flow shop production environments have been previously studied. Considering that the job shop production environment is one of the most important and practical production environments, in this research, this production environment has been considered and studied. Jia et al. (2002) studied the distributed scheduling problem for the first time. In their paper, a web-based system for production scheduling was proposed to facilitate cooperation between factories distributed in different geographical areas. They proposed a genetic algorithm (GA) to solve the problem. It includes one crossover gene and two mutation genes. In another research, Jia et al. (2003) suggested a modified genetic algorithm. The proposed algorithm has a two-step encoding method. To evaluate the performance of GA, they used the samples suggested by Muth et al. (1963). Chan et al. (2005) examined the problem raised by Jia et al. (2003). To solve this problem, they proposed a GA with dominated genes. To solve this problem, in another paper, Jia et al. (2007) proposed a GA and Gantt chart. The computational results showed that the proposed algorithm is efficient.

Moon and Seo (2005) proposed a mathematical model and an evolutionary algorithm (EA) to solve this problem. In their paper, flexible sequences, precedence constraints, and alternative machines are considered. Liu et al. (2006) hierarchically decomposed the multi-factory planning and scheduling problem into several levels of problems, including planning, scheduling, and material tracking feedback. To solve this problem in the dynamic environment, they proposed an approach based on a genetic algorithm, multi-agent method, user cooperation, and multi-closed loop control. Moon et al. (2008) used topological sorting

to create a set of operations sequences, and developed the evolutionary search approach to find possible solutions to the multi-factory scheduling problem in the job shop production environment. Wang et al. (2006) studied the problem of the dynamic distributed scheduling problem in the job shop production environment with the objective of minimizing production costs and delivery times. Chan and Chung (2007) proposed a GA with dominant genes for minimizing the makespan of the distributed job shop scheduling. Zhang et al. (2008) developed a multi-factory model for the distributed job shop scheduling problem. They proposed a distributed scheduling mechanism that combines the GA with a dynamic scheduling strategy. To solve the integrated scheduling and distribution problem in supply networks with different machines, Ławrynowicz (2008) proposed a metaheuristic algorithm. His/her hybrid approach is built upon the GA. In this problem, the author considered the constraints of priority and outsourcing and the sequences of different operations. Jeong and Yim (2009) investigated the problem of distributed scheduling in the job shop production environment with the objective of minimizing the makespan. Their study assumes that a production subsystem is responsible for maintaining its own private information and must work with others to achieve a global goal by sharing a minimum of private information. The authors proposed a Lagrangian relaxation method to solve the problem. Naderi and Azab (2014) proposed two mixed-integer programming models based on sequence and position variables for the distributed job shop production scheduling problem. To solve problems in medium- and large-size instances, they proposed three heuristic algorithms and three greedy heuristic algorithms. Also, Naderi and Azab (2015), in another paper, proposed a mixed-integer programming model with the objective of minimizing the makespan. They state that their proposed model is better than other proposed models in terms of size and complexity. To solve the problem, they developed a simulated annealing (SA) algorithm. Chaouch et al. (2017) proposed three algorithms named modified ant colony optimization for this problem. They showed that the modified ant colony optimization (MACO) algorithm has better performance than the other two algorithms. Xie et al. (2019) proposed a multi-objective artificial bee colony (MOABC) to solve the distributed job shop scheduling problem (DJSSP) with makespan and the total energy consumption criteria. Jiang et al. (2020) presented an effective modified multi-objective evolutionary algorithm with decomposition (MMOEA/D) to solve the energy-efficient distributed job shop scheduling problem (EEDJSP). In the proposed algorithm, they designed encoding and decoding schemes, several initialization rules, a collaborative search, and three problem-specific local intensification heuristics. Şahman (2021) proposed a discrete spotted hyena optimizer (DSHO) algorithm for this problem. In this research, the author compared the results of the proposed algorithm with four other heuristic algorithms. Wang et al. (2021) proposed an improved genetic algorithm (IGA) to solve this problem with the objective of minimizing the makespan. Fei et al. (2022) presented a mathematical model and simulation model for the joint distributed job-shop production and preventive maintenance scheduling. To solve the problem, they proposed the sequence exchange-based genetic algorithm. A summary of the reviewed papers is shown in Table 1.

According to Bagheri Rad and Behnamian (2022) and as shown in Table 1, all studies in the multi-factory job shop production system assume that all factories are owned by a central organization. Due to the globalization process in the field of production, factories can work together with independent ownership by creating a virtual production network. In this case, the factories can achieve more benefits than when they operate outside a network. In this problem, each factory may have a different objective function. Due to the mentioned reasons, in this study, it is assumed that the factories have independent ownership and are considered as single agents. Also, no research has been conducted considering a multi-agent approach with the arrival of jobs and the breakdown of machines in this problem. Most of the papers have focused on heuristic and metaheuristic algorithms, and a few have used hybrid algorithms. For this purpose, in this research, a hybrid algorithm based on the MA has been designed. In summary, the contributions of the current study compared to the reviewed research in the two areas of problem definition and solution approach are as follows:

- Problem definition: (i) Considering the transportation of jobs between factories, (ii) Considering the dynamic factors, including machine availability and the arrival of new jobs, (iii) Considering the virtual alliance between factories, (iv) Considering the multi-agent approach, (v) Considering resumable non-fixed policy for the machine breakdown, and, (vi) for the first time, studying a real and complex scheduling problem with assumptions close to the real world.
- Solution approach: (i) Developing a new mathematical model with several new constraints according to the assumptions under study, (ii) Designing a hybrid algorithm: Most of the papers in the literature review have focused on heuristic and metaheuristic algorithms and a few have used hybrid algorithms. So in this paper, a hybrid algorithm based on a non-dominated sorting genetic algorithm-II (NSGA-II) and local search has been proposed. This hybrid approach increases the speed of convergence, (iii) Introducing five well-defined problem-based neighborhood structures to increase the variety of solutions and a more comprehensive search within the

Table 1 Distributed scheduling with job shop environment

Paper	Multi-agent	Distributed network		Objective	Dynamic environment	Solving method
		Independent ownership	Interdependent ownership			
Jia et al. (2002)	–	–	*	Production costs	–	GA
Jia et al. (2003)	–	–	*	Makespan	–	GA
Chan et al. (2005)	–	–	*	Makespan	–	GA
Jia et al. (2007)	–	–	*	Makespan, production costs and tardiness	–	GA and Gantt chart
Moon and Seo (2005)	*	–	*	Makespan	–	EA
Liu et al. (2006)	–	–	*	Production costs	–	Hybrid algorithm
Moon et al. (2008)	*	–	*	Makespan	–	EA
Wang et al. (2006)	–	–	*	Production costs and delivery times	–	GA
Chan and Chung (2007)	–	–	*	Makespan	–	GA
Ławrynowicz (2008)	–	–	*	Makespan	–	Hybrid approach
Jeong and Yim (2009)	–	–	*	Makespan	–	Lagrangian relaxation
Naderi and Azab (2014)	*	–	*	Makespan	–	Heuristic
Naderi and Azab (2015)	*	–	*	Makespan	–	SA
Chaouch et al. (2017)	–	–	*	Makespan	–	MACO/AS/ACS
Xie et al. (2019)	*	–	*	Makespan and the total energy consumption	–	MOABC
Jiang et al. (2020)	–	–	*	Makespan and the total energy consumption	–	MMOEA/D
Şahman (2021)	*	–	*	Makespan	–	DSHO
Wang et al. (2021)	–	–	*	Makespan	–	IGA
Fei et al. (2022)	–	–	*	Makespan	–	GA
Present study	*	*	–	Makespan and the total energy consumption	*	MA

The sign "*" means to study the subject of the corresponding column in the research of the corresponding line

solution space, (iv) Proposing a new solution representation method, and (v) Embedding local search in the proposed algorithm.

3 Problem description and mathematical formulation

In this problem, there are F factories ($f=1,\ldots, F$) with different production speeds. Each of them has a job shop production system. Each factory is considered as an agent and seeks to minimize one of the objectives of total energy consumption or makespan. The symbols G_1 and G_2 are used to indicate the objectives of minimizing the makespan and total energy consumption of all factories. G_1 indicates factories that seek to minimize makespan. G_2 indicates factories that seek to minimize total energy consumption. Processing time of job j is $Ps_{j,i}$ and speed of machine i in factory f is v_{if}. When job j is processed in factory f, the time $Ps_{j,i}/V_{i,f}$ is needed to complete that job. If a job is not processed in its own factory, it needs tr_{fq} transfer time. In this state, job j will be processed in factory q instead of processing in factory f with time $2tr_{fq}+)Ps_{j,i}/V_{i,f}($. Also, N ($j=1, 2 \ldots N$) independent and non-priority jobs at zero time are ready to be processed and must be distributed among F factories. Each job contains a number of operations that must be processed on M machines ($i=1, 2\ldots M$) predefined in $p_{i,j,k}$ units of time. During the processing of jobs in the initial program, N' ($j'=1, 2\ldots N'$) new jobs may be entered. In this case, based on the manager's decision, the program is rescheduled at time Ts considering the not scheduled jobs of the original program and the new jobs. Also, machine i in factory f has maintenance and repair operation time $Pr_{r,i,f}$ which its completion time should be in the defined time window. Compared to classic job shop scheduling, in our problem, not only jobs should be scheduled in each factory, but also the assignment of the appropriate factory to the jobs and the schedule of maintenance and repair operation should be considered simultaneously.

The assumptions of this problem are as follows:

- At a time, each machine can process only one job and each job can be processed on only one machine**.**
- The processing time of jobs is deterministic.
- After starting the processing of the job on a machine, it must proceed on it without interruption, except when the machine breaks down.
- Each factory has a job shop production system.
- Loading and unloading times of the jobs are embedded in the transportation time.
- There are enough vehicles between factories to transport jobs.
- Setup time is considered at processing time.
- A resumable policy has been selected for periods of non-availability of machines due to breakdowns.
- Interruption of maintenance operations is not allowed.

3.1 Mathematical model

According to these assumptions, the index, parameters and decision variables are as follows.

3.1.1 Index

N: Number of jobs
m: Number of machines
F: Number of factories
f: Factory index ($f=1,2,..,F$)
i,l: Machine index ($i, l=1,2,\ldots,m$)
j_o: Initial jobs index ($j_o=1,\ldots,n$)
j_n: New jobs index ($j_n= n+1,\ldots,N$)
j,k: Sum of old and new jobs index ($j, k=1,2,\ldots,n+1,\ldots,N$)
r: Maintenance and repair operation index ($r=1,2,\ldots,R_{i,f}$)

3.1.2 Parameters

$R_{i,f}$: Total number of maintenance and repair activities on machine i in factory f
$Ps_{j,i}$: Processing time of job j on machine i
$v_{i,f}$: Speed of machine i in factory f
$tr_{f,q}$: Transfer time of job from factory f to factory q
L: A large number
$a_{j,i,l}$: 1, if machine i is used immediately after machine l in the process of processing job j; 0, otherwise
$w_{j,f}$: 1, if job jth is ordered to factory f; ; 0, otherwise
$ET_{r,i,f}$: The earliest time of completion of the maintenance and repair operation rth on machine i in factory f
$LT_{r,i,f}$: The latest time of completion of the maintenance and repair operation rth on machine i in factory f
$Pr_{r,i,f}$: Processing time of maintenance and repair operation rth on machine i in factory f
$[ET_{r,i,f}, LT_{r,i,f}]$: The time window to complete the maintenance and repair operation rth on machine i in factory f
$E^p_{j,i,f}$: Energy consumed of processing job j in factory f on machine i
$E^{idle}_{i,f}$: The idle energy consumed of machine ith in factory f
E_0: The common energy used by the general equipment of the factory
Ts: Rescheduling horizon

3.1.3 Variables

$X_{k,j,i}$: Binary variable; 1, if job j immediately is processed after job k on the machine i; 0, otherwise
$C_{j,i}$: Completion time of job j on the machine i
$y_{j,f}$: Binary variable; 1, if the job j is processed at f factory; 0, otherwise
$p_{j,i,f,q}$: Processing time of job j on machine i in factory q, if it was originally planned to be processed at factory f
$pt_{j,i}$: Processing time of job j on machine i considering the speed of machines and the movement of jobs between factories
$Cr_{r,i,f}$: Completion time the maintenance and repair operation rth on machine i in factory f
$yr_{r,j,i,f}$: Binary variable; 1, if job j is processed immediately after the maintenance and repair operation rth on machine i in factory f; 0, otherwise
$Ur_{r,j,i,f}$: Binary variable; 1, if the maintenance and repair operation rth is located between the processing time of job j on machine i is in factory f; 0, otherwise
Ec_w: Amount of energy consumption of processing job j in factory f on machine i
Ec_{idle}: The amount of idle energy consumption of machine i in factory f
$C_{\max}$, C'_{max}: Makespan
TEC: Total energy consumption (Kw)
The proposed model is as follows.

$$Z_1 = Min\, TEC \tag{1}$$

$$Z_2 = Min\, C_{\max} \tag{2}$$

$s.t.:$

$$\sum_{f=1}^{F} y_{j,f} = 1 \quad \forall j \tag{3}$$

$$p_{j,i,f,q} = w_{j,f}((Ps_{j,i}/v_{i,f}) + 2tr_{fq}) \quad \forall i,j,f,q \tag{4}$$

$$pt_{j,i} \geq p_{j,i,f,q} y_{j,q} - L(1 - y_{j,q}) \quad \forall i,j,f,q \tag{5}$$

$$C_{j,i} \geq pt_{j,i} + \sum_{r} Pr_{r,i,f} Ur_{r,j,i,f} \quad \forall i,j,f \tag{6}$$

$$C_{j,i} \geq C_{j,l} + pt_{j,i} + \sum_{r} Pr_{r,i,f} Ur_{r,j,i,f} \quad \forall j,f,i,l \neq i | a_{j,i,l=1} \tag{7}$$

$$C_{j,i} \geq C_{k,i} + pt_{j,i} - L\left(3 - X_{k,j,i} - y_{j,f} - y_{k,f}\right) + \sum_{r} Pr_{r,i,f} Ur_{r,j,i,f} \quad \forall j,i,f,k < N, j > k \tag{8}$$

$$C_{k,i} \geq C_{j,i} + pt_{k,i} - L\left(2 + X_{k,j,i} - y_{j,f} - y_{k,f}\right) + \sum_{r} Pr_{r,i,f} Ur_{r,j,i,f} \quad \forall j,i,f,k < N, j > k \tag{9}$$

$$C_{j,i} y_{j,f} \leq Cr_{r,i,f} - Pr_{r,i,f} + yr_{r,j,i,f} L + Ur_{r,j,i,f} L \quad \forall i,j,f,r \tag{10}$$

$$Cr_{r,i,f} \leq C_{j,i} y_{j,f} - pt_{j,i} + (1 - yr_{r,j,i,f}) L \quad \forall i,j,f,r \tag{11}$$

$$Et_{r,i,f} \leq Cr_{r,i,f} \leq Lt_{r,i,f} \quad \forall r,i,j \tag{12}$$

$$C_{\max} \geq C_{j,i} y_{j,f} \quad \forall i,j,f \in R_1 \tag{13}$$

$$C'_{\max} \geq C_{j,i} y_{j,f} \quad \forall i,j,f \in R_2 \tag{14}$$

$$Ec_w = \sum_{j=1}^{n} \sum_{i=1}^{m} \sum_{f=1}^{F} E^{p}_{j,i,f} pt_{j,i} y_{j,f} / 60 \quad f \in R_2 \tag{15}$$

$$Ec_{idle} = \sum_{i=1}^{m} \sum_{f=1}^{F} E^{idle}_{i,f} \left(C'^{f}_{\max} - \sum\nolimits_{j=1}^{n} pt_{j,i} y_{j,f} \right) / 60 \quad f \in R_2 \tag{16}$$

$$TEC = Ec_w + Ec_{idle} + (E_0 C'_{\max}) / 60 \quad f \in R_2 \tag{17}$$

$$C_{j,i} - pt_{j,i} - \sum_{r} Pr_{r,i,f} Ur_{r,j,i,f} \geq Ts \quad i,f,j \geq n+1 \tag{18}$$

$$C_{j,i}, pt_{j,i}, p_{j,i,f,q}, C_{\max}, C'_{\max}, Ec_w, Ec_{idle}, Cr_{r,i,f} \geq 0 \quad \forall i,j,f,r,q \tag{19}$$

$$Ur_{r,j,i,f}, yr_{r,j,i,f} \in \{0,1\} \quad \forall i,j,f,r \tag{20}$$

$$X_{k,j,l} \in \{0,1\} \quad \forall j < N, k > j, i \tag{21}$$

$$y_{j,f} \in \{0,1\} \quad \forall j,f \tag{22}$$

In the above model, Objective function (1) minimizes the total energy consumption and Objective function (2) minimizes the makespan. Constraint (3) determines each job must be assigned to a factory. Constraints (4) and (5) calculate the amount of processing time of each task considering the transfer times between factories when it is assigned to specific factories. Constraint (6) represents the time to complete a job is longer than the processing time and the maintenance and repair operation time. Constraint (7) indicates that each job should be processed just by one machine at a time. Constraints (8) and (9) represent one machine can process one job at a time. In other words, the two last constraints indicate that if the two jobs j and k are to be processed on machine i, job j on machine i can start processing when its previous job (k) on this machine is finished. Constraints (10) and (11) state that the processing jobs and maintenance and repair operations do not overlap. Constraint (12) ensures that maintenance and repair operations must be completed within a specified time window. Constraint (13) calculates the makespan of jobs belonging to the first agent. Constraint (14) calculates the makespan of jobs belonging to the second agent. Constraint (15) calculates the amount of machine energy consumption in the processing time of jobs. Constraint (16) calculates the amount of energy consumed in idle time. Constraint (17) represents the total energy consumption. Constraint (18) indicates the start time of new jobs must be after the rescheduling horizon. Model variables are defined by constraints (19–22).

3.2 Model solving

Since nonlinear models are more difficult to solve than linear ones, and our proposed model is nonlinear, a method for its linearization has been used. In this regard, suppose the variable $V = p.y$ is the multiplication of a binary variable (y) in a continuous variable (p). In this case, when the binary variable takes a value of one, the variable V will be equal to the value of the continuous variable; otherwise, it will take a value of zero. Therefore, Constraints (23), (24) and (25) are used to linearize these constraints (Glover and Woolsey 1974).

$$V_{j,i,f,q} \leq p_{j,i,f,q} \tag{23}$$

$$V_{j,i,f,q} \leq L y_{j,q} \tag{24}$$

$$V_{j,i,f,q} \geq p_{j,i,f,q} - L(1 - y_{j,q}) \tag{25}$$

The augmented epsilon constraint (AEC) method is one of the methods for solving multi-objective problems. This method reports a number of Pareto solutions resulting from the balance of objective functions (Mavrotas and Florios 2013). This approach is used in this research. The steps of this approach are shown in Fig. 1.

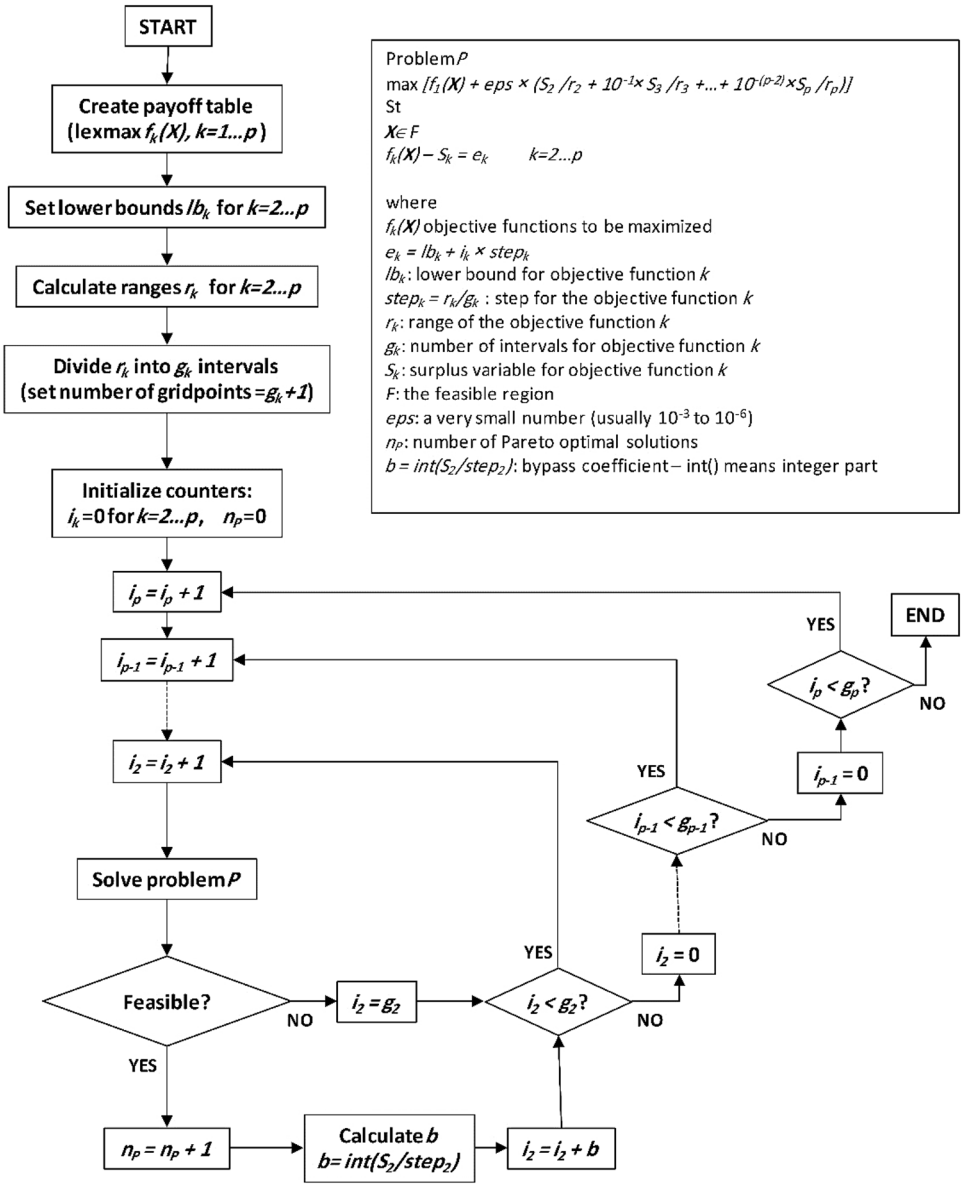


Fig. 1 Flow chart of augmented epsilon constraint method (Fei et al. 2022)

4 Solving approach

In a multi-factory job shop scheduling problem, if assumed that there is a single factory, the problem will become a classic scheduling problem (Behnamian and Fatemi Ghomi 2015). Since the classic job shop problem is Np-hard (Garey et al. 1976), the studied problem, which includes F factories, is at least as difficult as the single-factory problem, and therefore it is also Np-hard. On the other hand, in the distributed scheduling problem, in addition to the classical scheduling decisions, decisions must be made regarding the allocation of jobs to appropriate factories and this increases the complexity of the problem (Chung et al. 2009). Due to the complexities of this problem, it is impossible to solve it, especially in large-size instances, in a reasonable time. Therefore, here, to find good solutions in a reasonable time, metaheuristic algorithms are proposed. For this purpose, in this paper, a memetic-based non-dominated sorting genetic algorithm-II is proposed. Since the considered problem has two objectives, the proposed algorithm is designed based on the NSGA-II structure and is improved with several local searches.

4.1 Proposed memetic algorithm

The MA is a combination of one of the population-based algorithms and a local improvement method. In this research, due to the multi-objective nature of the problem, the NSGA-II structure has been used in the design of the proposed algorithm. The NSGA-II was proposed in 2002.

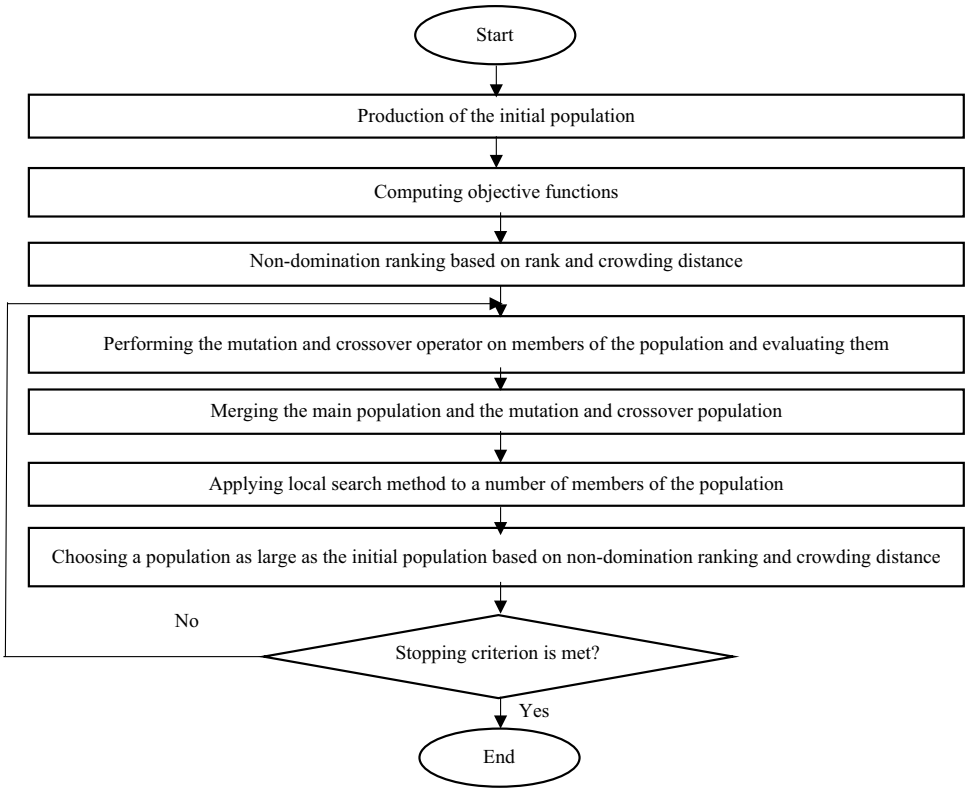


Fig. 2 Flowchart of the proposed algorithm

Initialization:
Define five neighborhood structures;
Select a number (n) of members of the population; (random)
For $i=1:n$
  Consider solution y_i;
  Apply Neighborhood structure 1 on y_i, (x_1)
  Apply Neighborhood structure 2 on x_1, (x_2)
  Apply Neighborhood structure 3 on x_2, (x_3)
  Apply Neighborhood structure 4 on x_3, (x_4)
  Apply Neighborhood structure 5 on x_4, (x_5)
end

Fig. 3 Pseudo-code of local search method

Job	1	2	1	2	3	1	2	3	3
Machine	1	2	2	1	1	3	3	3	2
Factory	1	2	1	2	2	1	2	2	2

Fig. 4 Solution representation

In this algorithm, the initial population is first generated, then all members of the population are ranked based on the domination rules. Then the crowding distance of the members with the same rank is calculated. In the next step, based on the binary tournament, parents are selected. In the binary tournament, the member with the highest rank is given priority. If two members have the same rank, the member with the greater crowding distance is given priority. In the next step, the mutation and crossover operators are used to generate the offspring. As shown in Fig. 2, the steps of the proposed MA are similar to the NSGA-II, except that at the end of each generation, a local search is performed on a number of populations to find better solutions. The steps of the local search method are shown in Fig. 3.

4.2 Implementation details

4.2.1 Solution representation

In this paper, matrix-based coding is used for solution representation. In this matrix, the jobs, machines and factories are in the first, second and third rows, respectively. The length of each row is equal to the total number of operations. The structure of the solution representation maintains the feasibility of each solution. Figure 4 shows an example with three jobs and two factories in which each job consists of three operations that must be processed on three machines.

4.2.2 Initial population

In the proposed algorithm, the initial population is generated randomly.

4.2.3 Crossover operator

In the proposed method, a single-point crossover operator is used.

4.2.4 Mutation operator

In this method, a pair swapping operator is used. In this method, two cells from the randomly selected row are selected and their locations are replaced with each other.

4.2.5 Local search

In the algorithm, five neighborhood structures were proposed as follows:

4.2.6 Neighborhood structure 1

In this structure, first, two columns are randomly selected, then the positions of the two operations (columns) are swapped together.

4.2.7 Neighborhood structure 2

In this structure, first, a random number *r* between *2* to *n* (vector length) is generated and then the priority of the two operations *r* and *r-1* are swapped.

4.2.8 Neighborhood structure 3

In this neighborhood structure, first, two jobs are randomly selected. Then, the location of the operation of the two jobs (not just an operation of the job) is changed one by one (peer to peer), according to the operation number. An example of this neighborhood structure is shown in Fig. 5. In this instance, jobs 1 and 3 are randomly selected and then their operation locations are moved together.

4.2.9 Neighborhood structure 4

In neighborhood structure 4, first, a factory with more workload is selected. Then, one of the remaining factories is randomly assigned to this job.

4.2.10 Neighborhood structure 5

In this neighborhood structure, first, two jobs (not just an operation of the job) from two different factories are selected. Then their factories are swapped. An example of the neighborhood structure is shown in Fig. 6. In this instance, jobs 1 and 2 are randomly selected and then their factories are swapped together.

4.3 Complexity of proposed memetic algorithm

The Time complexity of non-dominated sorting genetic algorithm-II (NSGA-II) for each generation is equal to the order $O(mN^2)$ (Jafari and Rezvani 2021). In this order, m is the number of objectives and N is population size. In this problem, there are two objectives ($m=2$). Since our proposed algorithm is based on NSGA-II, its $O()$is the same as NSGA-II.

5 Computational results

In this section, sample problems are defined to evaluate the performance of the proposed mathematical model and algorithm. The proposed mathematical model and algorithm are coded in GAMS 24.1.2/CPLEX SOLVER and MATLAB (R2011a), respectively. In small-size instances, validation of the proposed mathematical model has been done using GAMS software and the memetic-based non-dominated sorting genetic algorithm-II (MA). In order to validate the MA in large-size instances, a proposed algorithm entitled Hybrid Pareto-based Tabu Search Algorithm (HPTSA) by Li et al. (2018) is chosen. Furthermore, to evaluate and validate the proposed mathematical model and algorithm, all instances are generated randomly.

Job	4	2	**1**	4	**3**	**1**	2	**3**	**1**	2	4	**3**
Machine	1	2	**2**	2	**1**	**3**	3	**3**	**1**	1	3	**2**
Factory	2	2	**1**	2	**2**	**1**	2	**2**	**1**	2	2	**2**

Job	4	2	**3**	4	**1**	**3**	2	**1**	**3**	2	4	**1**
Machine	1	2	**1**	2	**2**	**3**	3	**3**	**2**	1	3	**1**
Factory	2	2	**2**	2	**1**	**2**	2	**1**	**2**	2	2	**1**

Fig. 5 Neighborhood structure 3

Fig. 6 Neighborhood structure 5

Job	1	2	1	2	3	1	2	3	3
Machine	1	2	2	1	1	3	3	3	2
Factory	1	2	1	2	2	1	2	2	2

Job	1	2	1	2	3	1	2	3	3
Machine	1	2	2	1	1	3	3	3	2
Factory	2	1	2	1	2	2	1	2	2

5.1 Performance evaluation metrics

To evaluate the efficiency of the solutions to multi-objective problems, various metrics have been proposed (Barma et al. 2022). In this research, three metrics of the mean ideal distance, diversification and runtime are used to evaluate the quality of results.

Mean ideal distance metric (MID): This metric calculates the closeness of Pareto solutions to the ideal point (Fattahi and Behnamian 2022). This metric calculates by Eq. (26), c_i in which the Euclidean distance of each member of the Pareto solutions from the ideal point (coordinate origin in the minimization), and n is the number of solutions in the Pareto layer. Obviously, the smaller values of this metric are more desirable.

$$\text{MID} = \sum_{i=1}^{n} c_i/n = \sum_{i=1}^{n} (f_{1i}^2 + f_{2i}^2 + f_{3i}^2 + \ldots)/n \tag{26}$$

Diversification metric (DM): This metric shows the diversification of the Pareto solutions. This metric is calculated using Eq. (27). The higher value of this metric is better.

$$\text{DM} = \sqrt{(\max f_{1i} - \min f_{1i})^2 + (\max f_{2i} - \min f_{2i})^2} \tag{27}$$

In this equation, f_{1i} and f_{2i} are the values of the first and second objective functions for solution i.

5.2 Parameter tuning

In this research, the Taguchi method is used for parameter tuning of the proposed algorithm. Taguchi method is simpler than other methods of design of experiments and also has less runtime. Effective factors in the performance of MA and their levels are shown in Table 2.

Note that the "local search rate" indicates what percentage of the population members enter the local search algorithm at the end of each iteration of the algorithm. According to the orthogonal arrays of the Taguchi method, the *L9* design has been selected as an experimental design for parameter setting for the proposed algorithm. Also, to achieve more reliable results, each combination is run ten times by the MA in Minitab16 software and their average is reported as the final output. The considered response variable is a composite index (Rahmati et al. 2013). The *S/N* ratio is calculated as Eq. (29).

Table 2 Parameters affecting the performance of the MA

Parameters	Sizes of problems		
	Small	Medium	Large
Population size	50	80	110
Mutation rate	0.1	0.15	0.3
Iteration	70	100	130
Local search rate	0.3	0.6	0.9

Table 3 Results of design of experiment of MA

Parameters	Sizes of problems		
	Small	Medium	Large
Population size	50	30	80
Mutation rate	0.1	0.1	0.1
Iteration	90	110	60
Local search rate	0.6	0.6	0.3

$$MOCV = MID/DM \tag{28}$$

$$\frac{S}{N}\text{Ratio} = -10\log\left(\frac{\text{sum}(y^2)}{n}\right) \tag{29}$$

The results of the Taguchi method for each level of factors are presented in Table 3.

5.3 Numerical results

In the small size, twenty instances were solved by the proposed algorithm and the ε-constraint approach. In order to increase the validity, each instance was run ten times by the proposed algorithm and HPTSA and the average results are reported in Table 4

Considering the values of performance criteria (MID, DM and Time), it is observed that the values obtained from the MA are close to the values of the ε-constraint approach. Also, with increasing the sizes of the instance, the runtime of the proposed algorithm is less than the ε-constraint approach. Due to the long runtime of the ε-constraint approach in GAMS software, the final instances in Table 4 were solved only by the MA. To illustrate the Pareto layer created by different methods in small-size instances, two instances (*i.e.,* instances 7 and 6) are considered. The obtained Pareto layers are shown in Fig. 7. As shown in the diagrams, the results of the MA and ε-constraint are very close to each other.

Due to the complex nature of the problem, large-size instances were solved by the proposed algorithm and the HPTSA. The obtained results are shown in Table 5.

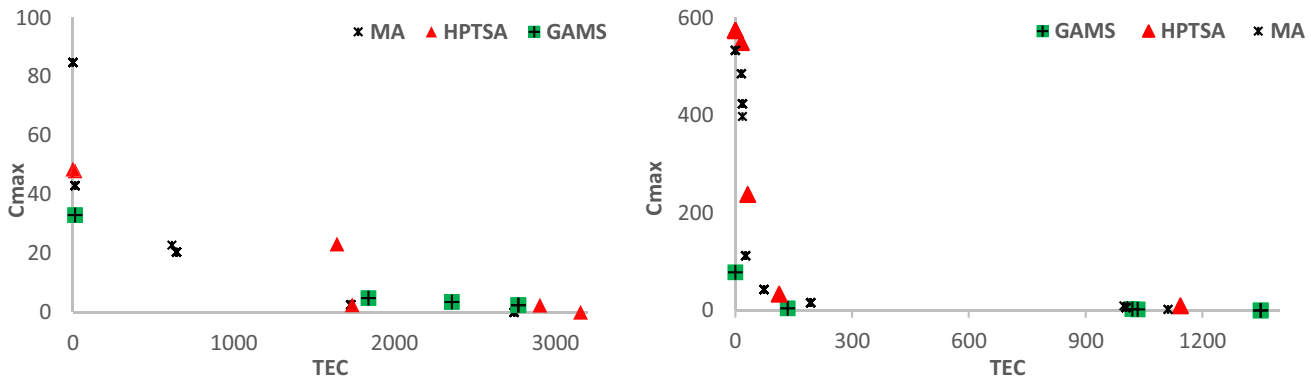
Relative Percentage Deviation (RPD) values of large-size instances are calculated by Eq. (30).

$$RPD = 100(\text{Algorithm}_{solution} - \text{Minimum}_{\text{solution}})/\text{Minimum}_{\text{solution}} \tag{30}$$

A diagram of average RPDs is plotted considering the MID and DM in Fig. 8a and b, respectively. As shown in Fig. 8a, the RPD values of the MID of the MA are lower

Table 4 Computational results the small-size instances

Instance	Number of machines	Number of jobs	Number of factories	GAMS (ε-constraint)			MA		
				MID	DM	Time (s)	MID	DM	Time (s)
1	3	2	2	174.727	329.785	3	142.5723	250.66	61.345
2	2	4	2	462.224	1153.698	5	455.6	935.138	65.541
3	2	7	5	621.114	776.994	13	632.418	1126.517	81.77
4	3	7	4	1178.573	2186.548	18	1130.62	1912.427	94.241
5	2	8	4	717.329	1788.442	32	691.203	680.515	94.767
6	3	8	4	827.855	1352.309	15	602.445	824.004	108.085
7	3	5	2	1649.065	2773.109	8	1652.528	2452.52	58.918
8	3	6	2	444.51	981.137	17	448.22	769.106	61.623
9	4	5	3	849.076	1681.255	10	864.727	1226.519	73.21
10	3	9	3	305.507	644.822	985	742.837	1331.372	160.832
11	2	12	2	407.998	1767.737	7380	577.434	1217.29	150.407
12	2	10	3	741.865	2094.851	4089	751.204	1010.167	87.21
13	2	12	5	417.471	791.887	5676	1642.231	2267.902	150.023
14	3	9	4	445.226	957.906	1396	989.878	1351.289	162.267
15	3	10	3	1415.606	2703.395	114	1422.287	1954.93	146.602
16	2	11	3	376.794	1101.065	9108	956.829	2037.502	149.511
17	3	15	4	–	–	–	1407.55	1946.998	223.353
18	2	16	3	–	–	–	1296.216	1777.359	163.03
19	3	12	4	–	–	–	932.119	1601.389	159.163
20	2	14	3	–	–	–	705.524	1163.338	160.402



(a) The Pareto layers of instance 7

(b) The Pareto layers of instance 6

Fig.7 Comparing the Pareto layer of algorithms by the ε-constraint approach

than the other algorithm, so the MA performs better. Also, as shown in Fig. 8b, the RPD values of the DM of the MA are less than the other algorithm. Since the higher DM, the better, and because the HPTSA has a higher DM, it has performed better.

To observe the effect of increasing the number of machines and factories on the performance of algorithms criteria, additional analyzes were performed and the results are shown in Fig. 9 a–e. As shown in Fig. 9a and c, with the increase in the number of machines and factories, the MA performs better than the HPTSA in terms of MID. Considering the DM, the solutions of HPTSA have a better dispersion than the MA. Also, it can be concluded from Fig. 9a that with an increase in the number of machines, the performance of both algorithms deteriorates in terms of the MID. With the increase in the number of factories, the performance of both algorithms improves in terms of the MID (see Fig. 9c). It can be concluded from Fig. 9b that with the increase in the number of machines, the performance of both algorithms improves in terms of the DM. Furthermore, as shown in Fig. 9d, with the increase in the number of factories, the efficiency of the MA decreases, and the

Table 5 Computational results in large-size instances

Instance	Number of machines	Number of jobs	Number of factories	MA			HPTSA		
				MID	DM	Time(s)	MID	DM	Time(s)
1	3	28	6	986.632	1077.638	471.664	1596.508	2073.985	541.16
2	3	33	6	1287.26	2191.308	518.994	2039.216	2986.581	774.862
3	3	41	7	5178.304	3595.417	673.801	7146.769	6532.587	769.283
4	3	70	5	6950.641	6251.801	1335.844	7289.787	7020.27	1527.389
5	3	100	5	15,461.5	7571.201	2567.84	18,037.51	9298.857	3380.624
6	4	22	5	4849.174	4699.319	397.903	8993.841	6684.329	516.231
7	4	30	7	4602.606	4123.887	592.451	5516.386	6291.122	774.139
8	4	36	6	4606.4	4879.362	715.916	6034.394	5938.058	786.569
9	4	45	7	7019.317	5830.58	875.588	10,477.22	10,019.84	956.62
10	4	70	5	11,802.5	8226.417	2011.819	15,302.52	10,691.85	2656.031
11	4	100	4	13,609	7725.494	3475.661	17,262.59	11,197	3330.509
12	5	21	5	2748.904	4046.511	420.042	4976.804	8665.081	776.039
13	5	30	4	7030.066	7870.675	797.984	7622.721	10,956.11	915.998
14	5	50	7	11,268.47	5843.734	1182.66	17,691.79	16,938.46	1159.184
15	5	100	6	18,569.98	7969.416	2418.648	24,250.46	15,502.56	2469.068
16	6	53	7	10,045.28	6473.085	1415.521	11,214.87	8773.551	1594.745
17	6	55	7	17,103.08	9024.009	1704.215	19,517.65	11,890.82	1821.746
18	6	70	5	14,099.99	10,337.63	2312.659	15,039.83	10,156.41	2389.503
19	6	100	5	29,438.2	15,846.43	4004.813	31,629.23	16,635.58	3038.007
20	6	150	6	33,711.64	18,754.78	4757.671	38,221.97	15,158.68	3826.063

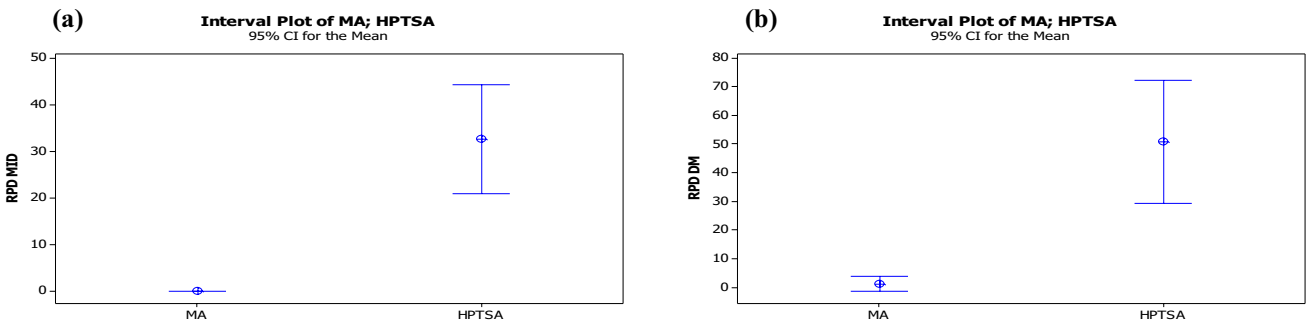


Fig. 8 **a** Diagram of average RPD of the MID for the MA and HPTSA algorithms in large-size instances**). b** Diagram of average RPD of the DM for the MA and HPTSA algorithms in large-size instances

efficiency of the HPTSA improves. Another important factor for comparing metaheuristic algorithms is their runtime. According to Fig. 9e, the runtimes of the algorithms increase with increasing dimensions of the instance. As shown in this figure, the runtimes of both algorithms are approximately close to each other.

Also, in this research, for comparing the capabilities of metaheuristic algorithms in large-size instances in finding good solutions in a reasonable time, they are statistically examined. In this regard, first, one of the basic analysis assumptions of variance (Normality of data) was examined. Shipro-Wilk test has been used to check the normality of the data (MID, DM, and run times). The test results of the normality of the algorithms according to the DM, MID, and run times are shown in Tables 6, 7, 8, respectively. The output of this test shows that the P-value of the MID, DM, and run times values of at least one of the two algorithms is less than 0.05. It can be concluded that the MID, DM, and run times values of the algorithms do not have a normal distribution. Therefore, the non-parametric Kruskal–Wallis (KW) test is used for statistical analysis. In this test, the Null-hypothesis is the equality of the average criterion of the proposed algorithm with the average criterion of the HPTSA. The results of the DM, MID and run times are shown in Tables 9, 10, 11, respectively.

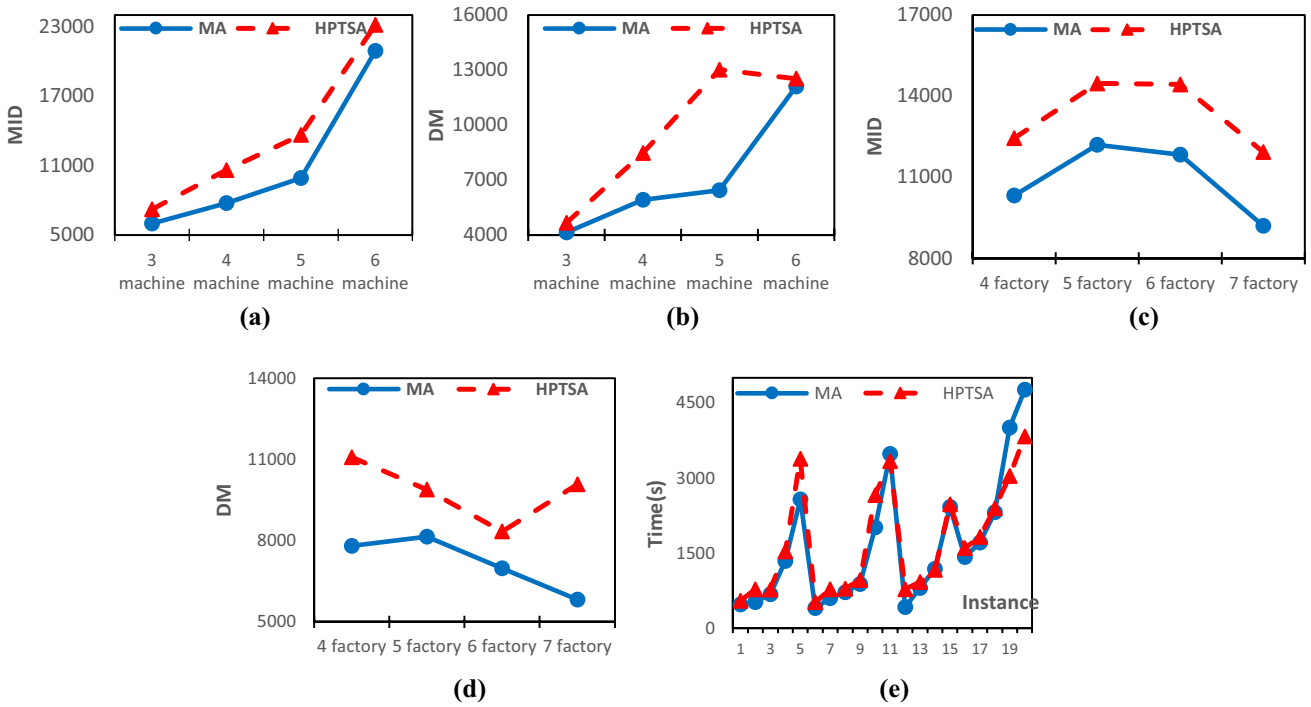


Fig. 9 Comparison of performance of algorithms in the large-size instances

Table 6 Shapiro–Wilk test results for the DM values of algorithms

	w	DF	p value
MA	0.878	20	0.01
HPTSA	0.961	20	0.558

Table 7 Shapiro–Wilk test results for the MID values of algorithms

	w	DF	p value
MA	0.877	20	0.016
HPTSA	0.906	20	0.053

Table 8 Shapiro–Wilk test results for run times of algorithms

	w	DF	p value
MA	0.861	20	0.008
HPTSA	0.872	20	0.013

Table 9 KW test results for the DM values of algorithms

KW chi-squared	25.3111
DF	1
p value	0.00000048

Table 10 KW test results for the MID values of algorithms

KW chi-squared	33.4405
DF	1
p value	0.0000000073

Table 11 KW test results for the run times of algorithms

KW chi-squared	0.3871
DF	1
p value	0.5338

As shown in the tables, the P-value of the two criteria of MID and DM of algorithms is less than 0.05, so the null hypothesis is rejected. Therefore, we conclude that there is a significant difference between the proposed MA and the HPTSA. According to the statistical test results and diagrams drawn in Fig. 9a–d, it can be concluded that the proposed algorithm performs better than the HPTSA considering the MID. Also, considering the DM, the HPTSA performs better than the proposed algorithm. Also, a statistical test has been used to compare the runtimes of the proposed algorithm with the HPTSA in large-size instances. As shown in Table 11, since the p value of the runtimes of the algorithms is more than 0.05, we conclude that there is no significant difference between the runtimes of the MA and the HPTSA.

The results of this study can be applied to the car manufacturing industry. More than 30,000 parts are needed to make a car. All the parts cannot be made in a single parts manufacturing factory. Thus in practice, multiple parts

manufacturing companies distributed in different locations constitute a virtual production network.

6 Conclusion and future research

In this paper, a distributed multi-agent job shop scheduling problem in the dynamic environment with the constraints of availability of machines and new job arrivals is studied. In this problem, independently owned factories are distributed in different geographical locations and each one is interested in minimizing one of the objective functions of total energy consumption or makespan. To solve the problem, first, a bi-objective mathematical model was developed. Then, due to the high complexity of the problem, a memetic-based non-dominated sorting genetic algorithm-II was proposed for solving large-size instances and improved by several local searches. To validate the proposed model and algorithm, the results of the proposed algorithm were compared to the ε-constraint approach. Also, to examine the performance of the MA on large-size instances, the results obtained from the proposed algorithm were compared to the HPTSA. For small-size instances, the results showed that the solutions obtained from the proposed algorithm are very close to the solutions of the ε-constraint approach. The difference between the proposed algorithm and the ε-constraint approach in small-size instances is that with increasing the size of the problem, the runtime of the proposed algorithm is less than the ε-constraint approach. Furthermore, large-size instances were statistically analyzed. The P-value of the criteria of MID and DM of algorithms was less than 0.05. Therefore, the proposed algorithm performs better than the HPTSA considering the MID. Also, the HPTSA performs better than the proposed algorithm, considering the DM. Therefore, in general, the large-size validation results showed that the proposed algorithm has a better performance than the HPTSA. There were three major limitations to this study. First, new performance metrics such as green scheduling and TBL could be studied. Note that the triple bottom line aims to measure the financial, social, and environmental performance of a company over time. The second is the lack of study of uncertainty in the problem. Failure to address solution methods based on the exact method is also a limitation of the current study. On the other hand, major limitations of the model include its resumable policy for periods of non-availability of machines, job shop production system, and embedment of loading/unloading times of jobs in the transportation time. Given the introduced limitations and shortcomings of current research, suggestions can be made for future research: (*i*) considering other objective functions, (*ii*) adding probability and uncertainty to the problem parameters, (*iii*) presenting exact algorithms, (*iv*) presenting other effective meta-heuristic algorithms, and (*v*) and our final suggestion is to add realistic assumptions such as considering loading/unloading times of jobs in the transportation time or changing the production workshop.

Availability of data and materials The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

References

Agnetis A, Billaut JC, Gawiejnowicz S, Pacciarelli D, Soukhal A (2014) Multiagent Scheduling: models and algorithms. Springer-Verlag, Berlin

Allaoui H, Artiba A (2006) Scheduling two-stage hybrid flow shop with availability constraints. Comput Oper Res 33(5):1399–1419

Bagheri Rad N, Behnamian J (2022) Recent trends in distributed production network scheduling problem. Artif Intell Rev 55:1945–2295

Barma PS, Dutta J, Mukherjee A et al (2022) A multi-objective ring star vehicle routing problem for perishable items. J Ambient Intell Humaniz Comput 13:2355–2380

Behnamian J, Fatemi Ghomi SMT (2015) Minimizing cost-related objective in synchronous scheduling of parallel factories in virtual production network. Appl Soft Comput 29:221–232

Chan FTS, Chung SH (2007) Distributed scheduling in multiple-factory production with machine maintenance. In: Wang L, Shen W (eds) process planning and scheduling for distributed manufacturing. Springer, London, pp 243–267

Chan FTS, Chung SH, Chan PLY (2005) An adaptive genetic algorithm with dominated genes for distributed scheduling problems. Expert Syst Appl 29(2):364–371

Chaouch I, Driss OB, Ghedira K (2017) A modified ant colony optimization algorithm for the distributed job shop scheduling problem. Proced Comput Sci 112:296–305

Chung SH, Chan FTS, Chan HK (2009) A modified genetic algorithm approach for scheduling of perfect maintenance in distributed production scheduling. Eng Appl Artif Intell 22(7):1005–1014

Fattahi Z, Behnamian J (2022) Location and transportation of intermodal hazmat considering equipment capacity and congestion impact: elastic method and sub-population genetic algorithm. Ann Oper Res 316:303–341

Fei Y, Ziqing L, Yuanjun L (2022) Simulation optimization on joint production and preventive maintenance scheduling for distributed job-shop. J Syst Simul 34(4):688–699

Garey MR, Johnson DS, Sethi R (1976) The complexity of flowshop and job shop scheduling. Math Oper Res 1(2):117–129

Glover F, Woolsey E (1974) Technical note—converting the 0–1 polynomial programming problem to a 0–1 linear program. Oper Res 22(1):180–182

Goli A, Babaee Tirkolaee E, Soltani M (2019) A robust just-in-time flow shop scheduling problem with outsourcing option on subcontractors. Prod Manuf Res 7(1):294–315

Goli A, Babaee Tirkolaee E, Aydın NS (2021) Fuzzy integrated cell formation and production scheduling considering automated guided vehicles and human factors. IEEE Trans Fuzzy Syst 29(12):3686–3695

Jafari V, Rezvani MH (2021) Joint optimization of energy consumption and time delay in IoT-fog-cloud computing environments using NSGA-II metaheuristic algorithm. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-021-03388-2

Jeong IJ, Yim SB (2009) A job shop distributed scheduling based on Lagrangian relaxation to minimise total completion time. Int J Prod Res 47:6783–6805

Jia HZ, Nee AYC, Fuh JYH, Zhang YF (2002) Web-based multi-functional scheduling system for a distributed manufacturing environment. Concurr Eng 10(1):27–39

Jia HZ, Nee AYC, Fuh JYH, Zhang YF (2003) A modified genetic algorithm for distributed scheduling problems. J Intell Manuf 14(3):351–362

Jia HZ, Fuh JYH, Nee AYC, Zhang YF (2007) Integration of genetic algorithm and Gantt chart for job shop scheduling in distributed manufacturing systems. Comput Ind Eng 53(2):313–320

Jiang ED, Wang L, Peng ZP (2020) Solving energy-efficient distributed job shop scheduling via multi-objective evolutionary algorithm with decomposition. Swarm Evol Comput 58:100745

Ławrynowicz A (2008) Integration of production planning and scheduling using an expert system and a genetic algorithm. J Oper Res Soc 59(4):455–463

Lee CY (1999) Two-machine flowshop scheduling with availability constraints. Eur J Oper Res 114(2):420–429

Li JQ, Duan P, Cao J, Lin XP, Han YY (2018) A hybrid Pareto-based tabu search for the distributed flexible job shop scheduling problem with E/T criteria. IEEE Access 6:58883–58897

Liu X, Bo H, Ma Y, Meng Q (2006) A new approach for planning and scheduling problems in hybrid distributed manufacturing execution system. World Congr Intell Control Autom 2:7357–7361

Lohmer J, Lasch R (2021) Production planning and scheduling in multi-factory production networks: a systematic literature review. Int J Prod Res 59(7):2028–2054

Mavrotas G, Florios K (2013) An improved version of the augmented eps-constraint method (AUGMECON2) for finding the exact Pareto set in multi-objective integer programming problems. Appl Math Comput 219(18):9652–9669

Moon C, Seo Y (2005) Evolutionary algorithm for advanced process planning and scheduling in a multi-plant. Comput Ind Eng 48(2):311–325

Moon C, Lee YH, Jeong CS, Yun Y (2008) Integrated process planning and scheduling in a supply chain. Comput Ind Eng 54(4):1048–1061

Muth JF, Thompson GL, Winters PR (1963) Industrial scheduling. Prentice-Hall, Englewood Cliffs

Naderi B, Azab A (2014) Modeling and heuristics for scheduling of distributed job shops. Expert Syst Appl 41(17):7754–7763

Naderi B, Azab A (2015) An improved model and novel simulated annealing for distributed job shop problems. Int J Adv Manuf Technol 81(1):693–703

Rahmati SHA, Hajipour V, Niaki STA (2013) A soft-computing Pareto-based meta-heuristic algorithm for a multi-objective multi-server facility location problem. Appl Soft Comput 13(4):1728–1740

Şahman MA (2021) A discrete spotted hyena optimizer for solving distributed job shop scheduling problems. Appl Soft Comput 106:107349

Wang S, Yu J (2010) An effective heuristic for flexible job-shop scheduling problem with maintenance activities. Comput Ind Eng 59(3):436–447

Wang Y, Yan L, Zhu H, Yin C (2006) A genetic algorithm for solving dynamic scheduling problems in distributed manufacturing systems. World Congr Intell Control Autom 2:7343–7347

Wang S, Li X, Gao L, Wang L (2021) An improved genetic algorithm for distributed job shop scheduling problem. In: Intelligent Computing Theories and Application, pp 37–47

Williams J (1981) Heuristic techniques for simultaneous scheduling of production and distribution in multi-echelon structures: theory and empirical Comparisons. Manage Sci 27:336–352

Xie J, Gao L, Pan QK, Tasgetiren MF (2019) An effective multi-objective artificial bee colony algorithm for energy efficient distributed job shop scheduling. Proced Manuf 39:1194–1203

Zhang Y-X, Li L, Wang H, Zhao Y-Y, Guo X, Meng C-H (2008) Approach to the distributed job shop scheduling based on multi-agent. In: 2008 IEEE international conference on automation and logistics, 2008, Qingdao, China, pp 2031–2034. https://doi.org/10.1109/ICAL.2008.4636496