International Journal of Innovative
Computing, Information and Control
Volume 11, Number 1, February 2015

ICIC International ©2015 ISSN 1349-4198
pp. 213–229

OPTIMIZING DISTRIBUTED PRODUCTION SCHEDULING PROBLEM IN FLEXIBLE MANUFACTURING SYSTEM SUBJECTS TO MACHINE MAINTENANCE: A MODIFIED CHEMICAL REACTION APPROACH

Mohd Nor Akmal Khalid and Umi Kalsom Yusof

School of Computer Sciences
Universiti Sains Malaysia
11800 USM, Penang, Malaysia
mnak104041@student.usm.my; umiyusof@cs.usm.my

Received October 2013; revised February 2014

Abstract. *As one of the many high-end industrial solutions, the flexible manufacturing system (FMS) has attracted many research initiatives over the years because of its competitiveness and rapid development. The recent trend in globalization has led to an egression of distributed, unique, and independent units of production centers. The FMS in a distributed system (FMSDS) is considered as a multi-factory environment, where jobs are processed by FMSs. Problems in FMSDS are related to the allocation of jobs to factories, independent assignment of job operation of machines, and operations sequencing on the machine. In addition, impact of the maintenance as one of the core parts of production scheduling has been previously neglected, thereby affecting the overall performance of the production schedule. Hence, this work considers maintenance as part of production scheduling. The objective of this study is to minimize the global makespan throughout the multi-factory network. This paper proposes a modified chemical reaction optimization algorithm to solve the problems in FMSDS. The molecule encoding explicitly represents the information on the factory, job, and maintenance; whereas a greedy decoding procedure exploits flexibility and determines job routing. An improvised crossover operator is used to improve the solutions by refining the most promising individuals of each generation. The proposed approach is compared with other algorithms, and algorithm performance is tested with several parameter refinements. Satisfactory results were achieved where about 1% to 26% deviations are observed when compared to other algorithms. The contribution of the study and the potential future works has also been discussed.*

Keywords: Production scheduling, Flexible manufacturing system, Distributed system, Machine maintenance, Chemical reaction optimization

1. **Introduction.** Production scheduling problems in the manufacturing sector have attracted research attention for several years. The rapid advancement of technology and the highly competitive market have introduced various alternatives to solve production scheduling problems. Exact approaches have become insufficient amidst the complex and challenging environment of the production scheduling floor. Production scheduling can be considered as an allocation problem in which a limited amount of resources (i.e., machines) are allocated to a number of tasks over a time horizon. Production scheduling problems are one of the *NP*-hard problems, and the feasible solutions for which are available in significant numbers for different task-resource assignments [1].

Competitive market demand and challenging manufacturing environment have changed the way organizations achieve success and competitive edge. The flexible manufacturing system (FMS) is the result of the growing demand for both quantity and quality such that the combination of the efficiency of high-production lines and the flexibility of job shops correspond well with mid-volume batch production and mid-variety of products [2]. Expensive equipments or machines that act as both resources and investment have indirectly increased the importance of effective and well-rounded performance as well as the efficient utilization of available resources. Extensive studies on FMS have been conducted since the 1980s. These studies are mostly focused on allocation, scheduling, loading, and control problems in FMS.

Generally, the single-factory production scheduling problem is concerned with the minimization of the total operating cost and completion time while fulfilling the orders of the machine assigned to process job operations. With the increasing globalization of market demand, the emergence of the distributed system (DS) has been significantly emphasized. A DS involves multi-factory production that is geographically distributed but remains effective in independently processing product parts. Each factory has unique production efficiency and constraints that depend on machine availability, labor costs and skill, and transportation facilities. These factors yield distinctive production lead times, operating costs, and completion times [3, 4]. Therefore, an exact solution to the production scheduling problem is difficult to establish, particularly because of the different process plan combinations in DS. Recent studies that consider these features of FMS have been reported [3, 5, 6]. With regard to the problem of the FMS in distributed systems (FMSDS), the optimization of the production schedule involves three hierarchical problems that need to be solved sequentially or simultaneously [7, 8]:

1. Allocation of the most suitable factory for the job (assignment problems).
2. Routing of the most suitable machine for each of the assigned operations of the job within the given factory (routing problem).
3. Sequencing the most suitable assignment of the operations to machines over the time span (sequencing problem).

In a real manufacturing environment, machine maintenance is unavoidable. Unexpected machine breakdown (stochastic unavailability) and scheduled preventive maintenance (deterministic unavailability) are the main causes of machine unavailability [9]. Machine preventive maintenance (PM) has attracted the attention of many researchers in the manufacturing domain because of its direct effects on production rate, product quality, machine availability and utilization ratio [4]. Nonexistent machine PM also disrupts the predetermined plan or scheduling because of process mismatching. Hence, considering PM in production scheduling plays a major role in perpetuating machine availability and utilization ratio while maximizing the facility with minimum cost and reducing unforeseen breakdown. To the best of our knowledge, the first work to address all features of FMSDS and consider PM is that of Chan *et al.* [4], which proposed the genetic algorithm with dominated genes.

2. **Related Work.** A series of studies on scheduling problems in FMS has been carried out. One of the classic solutions proposed for scheduling problems in FMS is the heuristic search algorithms. Several authors have proposed heuristic search or heuristic functions in solving the scheduling problems in FMS with respect to certain performance criteria [10, 11, 12, 13, 14, 15, 16, 17]. Most of the proposed solutions, which consider materials availability [10, 11], resource availability and its associated constraints [12, 13, 17], and the dynamic nature of the scheduling procedure [14], fail to integrate dynamic or reactive settings. Nevertheless, the consideration for dynamic and real-time scheduling environments has increased through the effort of Wang *et al.* [15] and Wang *et al.* [16]. Stochastic and unexpected events have also been considered in real scheduling environment.

Attempts have been made to solve scheduling problems in FMS by using artificial intelligence (AI)-based algorithms. Specifically, the classical AI approach has been used to solve scheduling problems in FMS [18, 19]. However, studies that consider machine availability, machine breakdown, or maintenance, and so on in solving dynamic scheduling problems in FMS are scarce. This gap in the literature is a result of the complexity of these scheduling problems and the various conflicting performance criteria [18]. Nevertheless, Lee [19] managed to propose a scheduling solution that can adapt to dynamically changing environments in the FMS through the continuous learning of historical data.

A larger portion of existing studies employ meta-heuristic algorithms that are nature-inspired or based on swarm intelligences or phenomenon mimicking. On the one hand, nature-inspired algorithms can be defined as algorithms derived from natural behaviors–from behaviors or processes of molecular reactions to complex cortical maps of the biological organization [20]. The artificiality represented in biological processes has inspired researchers to employ various computing optimization algorithms, such as genetic algorithm (GA) [21], simulated annealing [22, 23], shuffled frog leaping algorithm [24], and symbiotic evolutionary algorithm [25]. On the other hand, swarm intelligence or "collective" intelligence refers to the decentralized and self-organized problem-solving behavior, which is derived from the interactions of individual agents with other agents, in reacting to the local environments. Examples of such algorithms include ant colony optimization (ACO) [26], particle swarm optimization (PSO) [27], artificial immune system (AIS) [28], artificial bee colony (ABC) [29], and the recently adopted biogeography-based optimization (BBO) [30] and cuckoo search (CS) [31]. Another rare derivation of meta-heuristic algorithms is the algorithm that mimics a certain natural phenomenon. This phenomenon-mimicking algorithm refers to the optimization processes conducted through the emulations of naturally occurring phenomenon. Examples include the harmony search algorithm [32] that mimics the improvisation process of a musical performance and tabu search [33] that imitates the phenomena of accursed or "taboo" belief of the search process behavior.

Although certain limitations have been identified in the utilization of meta-heuristics algorithms to solve the scheduling problems in FMS [34], the efforts to employ these algorithms have been continuous in the last 25 years. Some existing studies focus on static scheduling environments in FMS with either single [25, 31, 35, 36, 37, 38, 39] or multiple [40, 41] performance criteria. In addition, the implementation of the meta-heuristics with respect to the problem domain tends to be arduous. However, attention toward dynamic scheduling environments has increased because of the importance of reducing scheduling time [42], the difficulty in scheduling and the short validity of implementations [43], and the need to enhance productivity by incorporating alternative scheduling plans or routing [34, 38].

Proposed in 2010 by Lam and Li [44], the chemical reaction optimization (CRO) algorithm is fairly new in optimization domains. The CRO algorithm is based on the free-form molecule behavior in a container that is characterized in the theory of conservation of energy. Such behavior intensifies through on-wall ineffective collision and inter-molecular ineffective collision while balancing the diversification features through decomposition and synthesis operators that renown for relative escapes in the local optimum. Given the fairly stable convergence rate of the CRO algorithm, global optimum can be achieved quickly. The promising results of CRO have increased the number of studies that employ the CRO algorithms to solve various problem domains. Sun *et al.* [45] proposed a hybrid CRO with the Lin-Kernighan search to solve the well-known travelling salesman problem, whereas Truong *et al.* [46] proposed a hybrid CRO with a greedy strategy algorithm to solve the 0-1 knapsack problem. Alatas [47] proposed a modified version of CRO, the efficiency of which was tested on a well-known benchmark mathematical function. To solve the same test problem, Yang *et al.* [48] also proposed a modified CRO (MCRO) that incorporates the global-best solution information into the search equation and thus improves the exploitation strength of the original CRO. CRO has also been adopted in many assignment and scheduling problems [44, 49, 50, 51].

Several studies have dealt with scheduling problem in FMS through bio-inspired population-based meta-heuristic algorithms. These algorithms have evidently been reported as prominent and dominant in most optimization domains. To the best of our knowledge, the first work to address a similar problem under consideration using the CRO algorithm is that of Li and Pan [51]. However, the CRO algorithm is rarely employed in the manufacturing domain, particularly in solving the scheduling problem. Such rarity introduces a great opportunity to pioneer this algorithm. In addition, CRO features which are capable of escaping the local optimum and exploring diverse solution space to achieve global optimum serve as a potent motivation to undertake CRO as a solution for the underlying problems in FMSDS subject to maintenance. The objectives of this study is to propose an MCRO algorithm with guided initialization mechanism to yield optimal makespan for a production scheduling plan while considering the impact of maintenance inclusion.

3. **The FMSDS Problem.** The problem in FMSDS can be stated as follows: a number of jobs (i) are expected to be received in the distributed network, and a suitable factory ($f = 1, \ldots, F$) will be assigned to the jobs such that a corresponding production schedule is generated. Each individual factory has a number of machines ($h = 1, 2, \ldots, H_f$) with varying efficiencies or operating lead times (T_{ijfh}) in producing various product types. Each job has up to N_i operations, and every operation can be performed in more than one machine (not all) in the same factory. The travel time between factory f and job i is denoted as D_{if}.

Each machine conforms to a maximum machine age (M), which is equal to the cumulated processing time of operations. As outlined in Chan *et al.* [4], a maintenance procedure must be carried out right after the completion of the current operation when the machine age reaches the threshold denoted as M. After every maintenance, the machine age of the particular machine is reset to 0.

The objective of the study is to minimize the total maximum makespan of the last job operation. The objective function is defined in Equation (1). Completion time (C_i) is defined as the summation of the completion time of the last operation N_i of job i and the delivery time between the factory f and the job i, as defined in Equation (2). The decision variables are as follows: χ_{ij} which is denoted true if job i is allocated to factory f; δ_{ijfhk}, which is denoted true if operation j of job i occupies time slot k on machine h in factory f; and γ_{ijfh}, which is denoted true if machine h in factory f is maintained after operation j of job i; Once the decision variable is determined, the starting time value of operation j of job i (S_{ij}), the ending time of operation j of job i (E_{ij}), and completion time (C_i) can be calculated.

$$\text{Objective } Z : \min(\max\{C_i\}). \tag{1}$$

$$C_i = E_{iN_i} + \sum D_{if}\chi_{if}. \tag{2}$$

The problem is subject to the following constraints:

1. Every operation can only begin after the completion of the prior operation.
2. An operation continues until it finishes without any disruption.
3. The assigned time slot must be equal to the required operation time.
4. Each operation must be carried out on a single machine throughout the horizon.
5. Each operation must be executed on a single machine at each unit of time.
6. Each machine must handle only a single operation at each unit of time.
7. Each job can only be assigned to a single factory.

4. CRO Algorithm for FMSDS.

4.1. Generic CRO. The CRO algorithm loosely mimics what happens to molecules in a chemical reaction system within a closed container. A chemical system undergoes a chemical reaction when it is unstable, that is, when it possesses excessive energy. The system manipulates itself to release the excessive energy and consequently achieve stability. This manipulation is called chemical reaction. The molecules represent the solution for the considered problem and are characterized by several properties. A molecule is composed of several atoms and is characterized by the individual properties of such atoms (i.e., atom type, bond length, angle, and torsion) [44]. Any change in the atom characteristic will distinguish the molecules. As such, changes in molecular structure are tantamount to switching to another feasible solution. Each molecule possesses two kinds of energy, namely, potential energy (PE) and kinetic energy (KE). PE corresponds to the objective function of a molecule, whereas the KE of a molecule represents its ability to escape from a local minimum.

The CRO algorithm is governed by the two fundamental laws of thermodynamics: conservation of energy and state of equilibrium. The first one states that an energy can neither be created nor destroyed; energy can be transformed from one form to another or from one entity to another. Each chemical substance possesses PE and KE, whereas the surrounding energy is symbolically represented by a central energy buffer. The second law states that the entropy of the system tends to increase; entropy refers to the degree of disorder measure. All reacting systems tend to reach a state of equilibrium, in which their potential energy drops to a minimum. In CRO, this phenomenon is depicted in the conversion of PE to KE and by gradually losing (KE_{loss}) of the energy of the chemical molecules to the surroundings (buffer).

The basic CRO involves four elementary reactions: on-wall ineffective collision, decomposition, inter-molecular ineffective collision, and synthesis. These elementary reactions can be categorized into single molecular reactions and multiple molecular reactions. The on-wall ineffective collision and decomposition reactions are single molecular reactions, whereas the inter-molecular ineffective collision and synthesis reactions are multiple molecular reactions.

- The on-wall ineffective collision reaction occurs when a molecule hits the wall and then bounces back. Some attributes of the molecule (ω) change after the on-wall collision. In this case, the molecule becomes a new molecule (ω') if the given condition is satisfied. After the on-wall ineffective collision,the molecule ω loses a portion of KE to the buffer. By losing KE to the environment, the molecule can improve convergence and local search abilities.
- The decomposition reaction is used to mimic the process in which the molecule hits the wall and then decomposes into two or more pieces. Two situations should be considered for the decomposition reaction: (1) the molecule has enough energy to complete the decomposition; (2) otherwise, the molecule should obtain energy from the energy buffer.
- The intermolecular ineffective collision is the process in which two or more molecules share information and then produce another set of two or more molecules. This reaction mimics the process in which two molecules collide with each other and then bounce away.
- The synthesis reaction is the process in which more than one molecule collides and combines together. Suppose two molecules $\omega 1$ and $\omega 2$ collide with each other. A new molecule ω' is then produced.

4.2. **Proposed MCRO.**

4.2.1. *Antibody encoding and decoding.* The information encoded in the molecule of the MCRO algorithm for FMSDS must specify the allocation of each job to the factory, the routing of every job through the machine, and the sequence of the operations. This encoding scheme follows the simple operation-based encoding method proposed by Jia *et al.* [7] for distributed scheduling problems without routing flexibility. Relevant extension that includes the flexibility issues of FMSDS is considered in the encoding scheme. The size of an atom ($atom_p$) in a single molecule is equal to the total number of operations of all the jobs. Each molecule is represented by a triplet notation (f, i, p), where (f) represents the factory, (i) represents the assigned job, and (p) represents the PM flag. Note that all the operations of the same job are represented by different atoms within the same molecular structure. An operation is interpreted according to the order of atom occurrence on the molecule, given that the order for the operation of a job is fixed. Following the adoption of a simple representation by Jia *et al.* [7], no information about alternative machine routes is explicitly encoded into the atom. This information will be retrieved during the decoding phase. A sample individual is given in Figure 1.

1, 3, 0	**1, 3, 0**	**1, 2, 1**	**2, 1, 0**	**1, 3, 1**	**1, 2, 0**	**2, 1, 1**

FIGURE 1. A sample molecular encoding

Assume job 1, job 2, and job 3 have two, two, and three operations, respectively, such that a molecule consists of seven atoms. Each atom consists of three types: “2, 1, $< p >$,” “1, 2, $< p >$,” and “1, 3, $< p >$.” In this case, jobs N_1 and N_2 are processed in factory F_1, and N_3 is processed in factory F_2.

In the decoding process, the information provided by each molecule is exploited to generate a schedule plan in which the PE of each molecule is evaluated. The objective of the FMSDS is to minimize the global makespan of the factory network so that the affinity of an individual is inversely related to the global makespan.

As previously discussed, molecules explicitly represent information on job assignments to factories, and the order of the atomic structure is relevant in determining the priority of each operation without consideration of the information on job routing. To simplify the encoding scheme, the flexibility problem is considered in the decoding phase, during which the scheme can dispatch job operations to one of the alternative machines of the selected factory. The information on job routing is thus implicitly encoded in the decoding process. Based on the order determined by the molecular structure, operations are considered sequentially. When the respective operation is dispatched to a machine, the starting time equals the completion time of the last operation assigned to the machine. If the considered operation requires more than one machine, the decoding process selects the routing that guarantees the lowest current local makespan; the machine that completes all assigned operations in the shortest time is chosen. If different routings lead to the same current makespan, then the machine with the smallest processing time is chosen. If the available machines have the same smallest current makespan and processing time, any of them is randomly selected to give the optimization algorithm the opportunity to search different regions of the solution space. The decoding process is completed by adding the delivery time (according to the factory where the job is assigned) as soon as all the operations have been scheduled, thus obtaining the local and global makespans.

4.2.2. *Population initialization.* The initial population is determined by three phases. In the first phase, jobs are randomly generated until all the operations of the jobs are generated. In the second phase, jobs are assigned randomly to factories, and the related operations of the jobs are amended to satisfy the factory allocation constraints. In the final phase, the maintenance flag is generated randomly. This process is repeated until all individuals of the population (pop_N) are initialized.

Each molecule is initialized with an initial KE of 10000, which is enough energy for each molecule to accelerate the CRO processes. After population initialization, each molecule is evaluated to update its PE value. Given the variable populations of CRO, a minimum limitation $pop_N \geq 3$ is imposed, and a relatively small pop_N is initialized. The KE loss rate KE_{loss} is defined as the amount of KE lost by an individual molecule every time an elementary reaction of CRO occurs. The buffer is also initialized with a value of zero.

4.2.3. *The elementary reactions.* In performing the elementary reactions of CRO, a probabilistic p_{coll} value is used to determine whether a uni-molecular or inter-molecular reaction occurs. If a randomly generated value $rand \geq p_{coll}$, then an inter-molecular reaction occurs; otherwise, a uni-molecular reaction occurs. In the case of a uni-molecular reaction, a randomly selected molecule ω is considered for on-wall ineffective collision, which is modified by the mechanism in Section 4.1 to produce a new updated molecule ω'. Molecule ω' is considered as a replacement for molecule ω provided that it satisfies the following:

$$PE_\omega + KE_\omega \geq PE_{\omega'} \tag{3}$$

If the condition in Equation (3) is not satisfied, then decomposition reaction occurs. That is, molecule ω is modified based on the mechanism in Section 4.1, whereas molecule ω' is considered as the two new molecules produced by the decomposition operator. Reciprocally, the two molecules are selected randomly to perform the mechanism in Section 4.1. Consequently, two molecules ω_1' and ω_2' are produced. The two molecules replace molecule ω given that the following condition is satisfied:

$$PE_\omega + KE_\omega \geq PE_{\omega_1'} + PE_{\omega_2'} \tag{4}$$

In the case of inter-molecular reaction, two molecules ω_1 and ω_2 are selected randomly to perform inter-molecular ineffective collision. These two new molecules will replace the two selected molecules (ω_1 and ω_2) if the condition given below is satisfied:

$$PE_{\omega_1} + KE_{\omega_1} + PE_{\omega_2} + KE_{\omega_2} \geq PE_{\omega_1'} + PE_{\omega_2'} \tag{5}$$

If the condition in Equation (5) is not met, synthesis reaction occurs. That is, two molecules ω_1 and ω_2 are randomly selected to perform the synthesis mechanism in Section 4.1, in which two molecules are produced. The best molecule is selected as the new molecule ω'. This new molecule ω' is considered in the population, whereas the two molecules ω_1 and ω_2 are removed, given that the following condition is satisfied:

$$PE_{\omega_1} + KE_{\omega_1} + PE_{\omega_2} + KE_{\omega_2} \geq PE_{\omega'} \tag{6}$$

Based on all the elementary reactions above, a greedy algorithm is embedded to maintain the performance of the algorithms in producing effective solutions and to enhance the convergence rate. If any of the conditions in Equations (3), (4) and (5) are satisfied; the solution must be satisfactory such that molecule $\omega' > \omega$. When more than one solution is available, either one of the two solutions must satisfy the aforementioned conditions.

4.2.4. *Modified ineffective collisions.* Similar to the mutation operator in the GA, the on-wall ineffective collision via the MCRO algorithm in the canonical CRO generates a new neighboring molecule from a given one. In this study, several neighboring approaches are used in the on-wall ineffective collision function to improve the exploitation capability of the algorithm. Given molecule ω, two positions r_1 and r_2, where $1 \leq r_1 < r_2 \leq atom_p$ are randomly generated. The neighboring approaches are described as follows:

- End-to-end swapping mechanism (EESM), which involves the reversal of each element between the first and the last atomic structure of $atom_p$;
- Simple swapping mechanism (SSM), which involves the swapping of the two atomic structures at r_1 and r_2.

The two approaches are given in Figures 2 and 3, respectively. However, the probability of performing the above mechanisms is equalized. If the condition of ineffective collision is true, then the probability of selecting between the EESM and SSM is 0.5. The high exploitation level of the MCRO algorithm is utilized to give good molecules several chances to manipulate their atomic structures while allowing the less promising ones to participate in the iteration. In other words, the search moves towards promising regions while guaranteeing the diversity of the solution candidates and preventing a premature convergence of the method.

In the case of multiple factories and maintenance considerations, the following atomic twitching mechanisms are conducted: random factory assignment (RFA) and random scheduled maintenance (RSM). These mechanisms are aimed at exploring several solutions in the search space with different assignments of jobs to factories and varied scheduled maintenance. Note that to maintain consistency in the atomic structures of the remaining molecules and to meet the factory constraint, all atomic structures must reflect the new job assignments; that is, all atomic structures related to the selected job in the molecule must be updated. This condition is demonstrated in Figure 4.

4.2.5. *Modified synthesis.* The synthesis reaction is used to produce a single molecule by combining two or more molecules. In the proposed MCRO, the crossover function is embedded in the synthesis process. The synthesis reaction is realized in three phases.

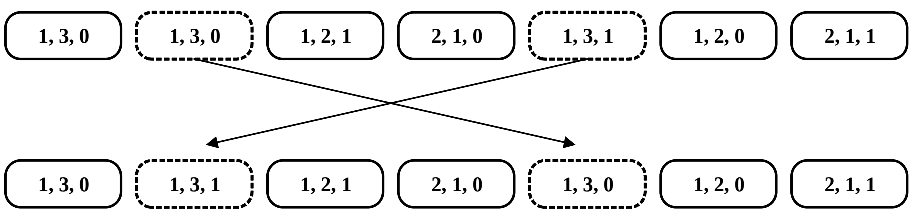

FIGURE 2. An illustration of SSM

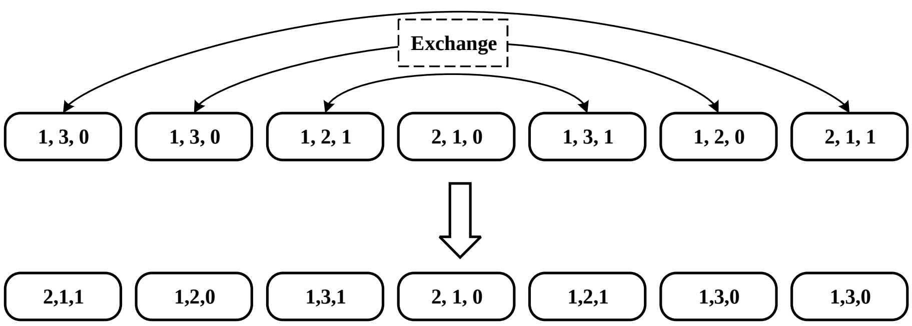

FIGURE 3. An illustration of EESM

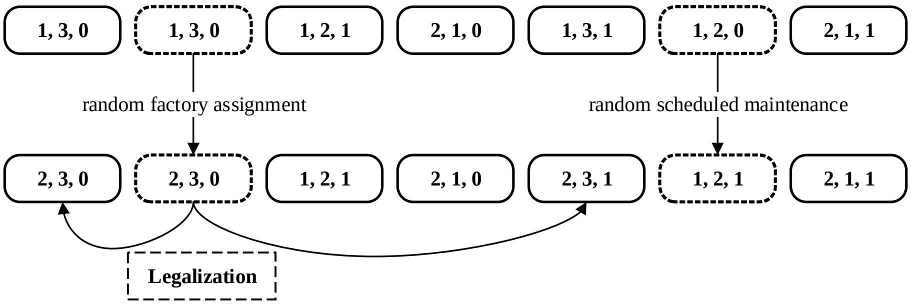

FIGURE 4. Atomic twitching: RFA and RSM

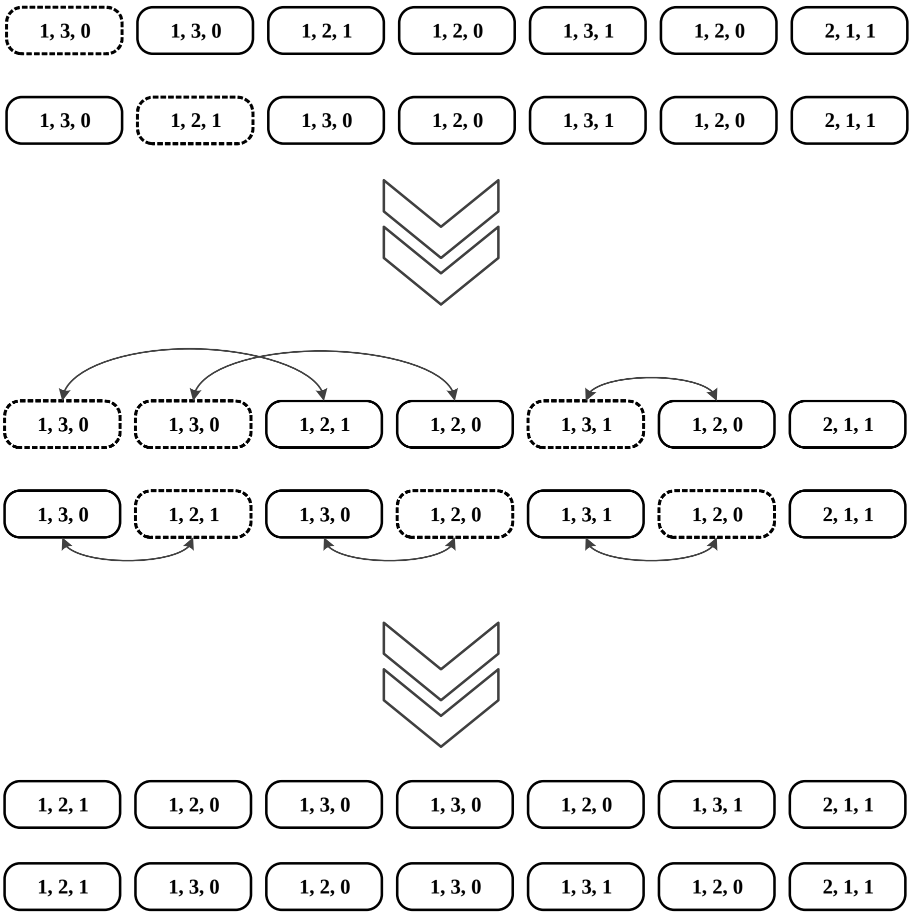

FIGURE 5. Illustration of TX

First, two molecules ω_1 and ω_2 are selected from the current population pop_N. Second, the crossover operator is applied to the two selected molecules to produce $\omega_1^{'}$ and $\omega_2^{'}$. Third, the best child molecule is selected as the new molecule $\omega^{'}$. The selection is carried out by comparing ω_1 and ω_2 with $\omega_1^{'}$ and $\omega_2^{'}$ and assessing whether any of the following conditions are satisfied: $\omega_1^{'} < \omega_1$, $\omega_2^{'} < \omega_2$, $\omega_1^{'} < \omega_2$ or $\omega_2^{'} < \omega_1$.

The crossover operator employed is similar to that in GA. The information encoded between parent molecules is exchanged to produce two child molecules. However, the crossover operator in the proposed MCRO is the trajectory crossover (TX) introduced by Rodriguez-Tello *et al.* [52] because of the encoding scheme limitation of the molecule. TX generates a new child molecule while exploring trajectories that connect two parents (ω_1 and ω_2). Starting from one parent, called the initial solution, a trajectory in the neighborhood space is produced and serves as a guide toward the alternate parent, called the guiding solution. This process is achieved by choosing moves that introduce attributes contained in the guiding solution; each new solution in the trajectory corresponds to an individual.

In the TX operator, the child inherits any atomic structures common to both parents. Starting at a random position of the parents, their atomic structures are examined from left to right in a cyclic manner. If the elements at the position being examined are the same, that position is skipped; otherwise, a swap is performed between two elements in parent ω_1 or in parent ω_2, whichever produces the best solution. In this way, the atomic solutions at the analyzed position become alike. This process is repeated until all atomic positions have been considered. All molecules obtained using this process are the valid offspring of ω_1 and ω_2; the best child molecule ω' among all molecules is returned. This process is illustrated as Figure 5.

4.2.6. *Termination condition.* The termination condition is based on the number of iteration $Iter_N$ used. Given the relatively large and unknown limitation of the dataset in this problem, neither the time limit nor the objective limit is used respectively. The maximum number of iterations used in this study is set to 5000, which is large enough to achieve a feasible solution and to conduct comparative analyses. Figure 6 shows the overall flowchart of the proposed MCRO algorithm.

5. **Computational Results.** The performance of the MCRO was tested under different settings. Four datasets were considered. The first, second, and third datasets were

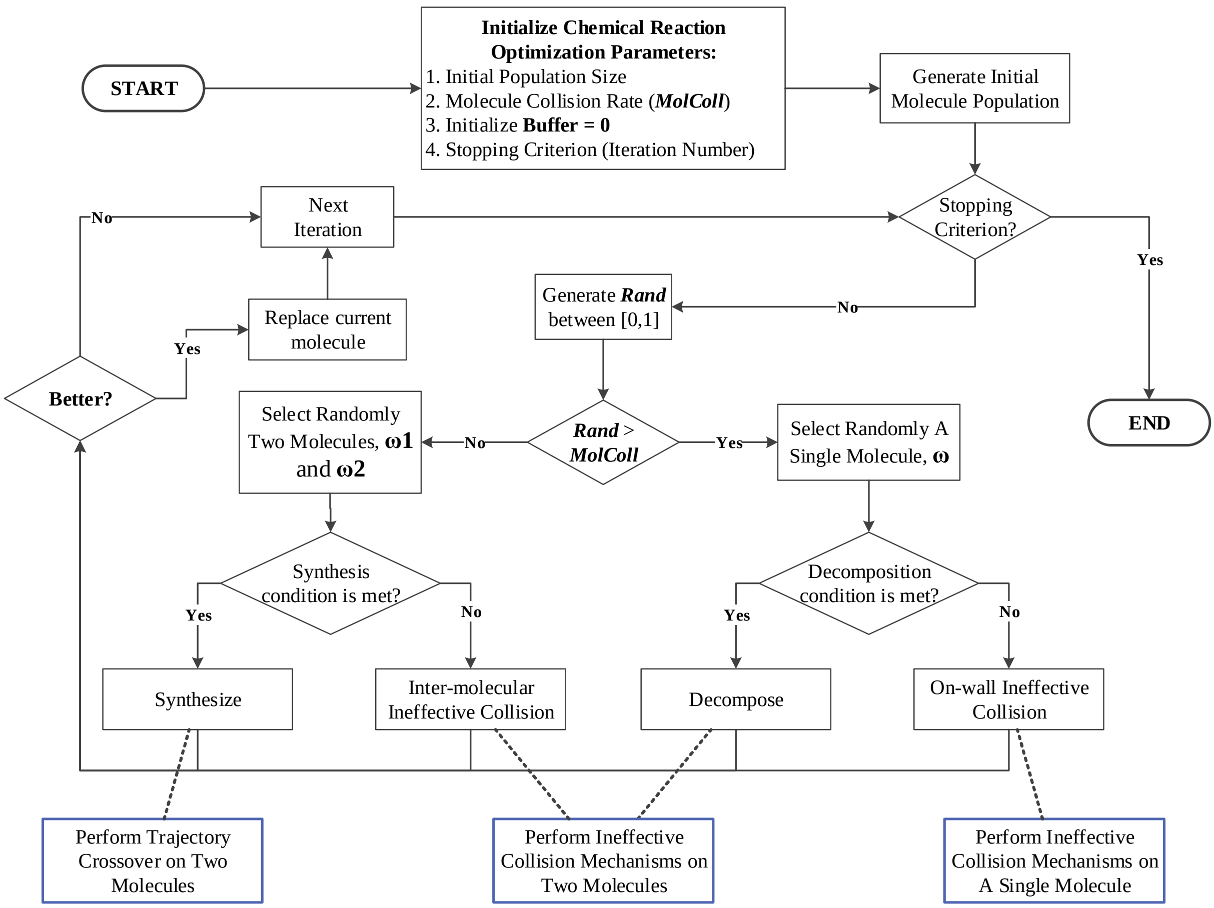

FIGURE 6. Flowchart of the proposed MCRO

obtained from Chan *et al.* [4, 53, 54], whereas the fourth dataset was obtained from the benchmark data of Fisher and Thomphson [55]. Two separate experiments were conducted. The first experiment used the first, second, and third datasets, whereas the second experiment used the fourth dataset. The first experiment compared the MCRO with other algorithms designed for FMSDS, namely ACO by Kumar *et al.* [38], GADG by Chan *et al.* [4, 53, 54], modified GA with dominant gene by Chung *et al.* [56], and improved GA (IGA) by De Giovanni and Pezzella [8]. The second experiment compared MCRO with other algorithms that were employed in the same dataset; the algorithms were modified GA by Jia *et al.* [7] and IGA by De Giovanni and Pezzella [8]. MCRO was implemented in C# compiler and ran independently on a personal computer equipped with a 2.0 GHz Intel Core i5 processor and 2 GB RAM.

All the datasets considered in this study are summarized in Table 1. MCRO parameters were calibrated for the preliminary test on all datasets described above. The settings of the four parameter options for each datasets considered are given in Table 2.

Results of the first and second experiments are given in Table 3. The first column reports the dataset name of testing instance, and the following column represents the compared algorithms consecutively with the relative deviation of makespan with respect to the proposed MCRO. The relative deviation is defined as in Equation (7).

$$dev = [(MK_{comp} - MK_{IIA})/MK_{comp}] * 100\% \tag{7}$$

MK_{IIA} is the makespan obtained by the proposed MCRO, and MK_{comp} is the other algorithm that was presented for comparison. As given in Table 3, MCRO outperforms other algorithms by obtaining optimal results for most datasets in both experiments considered in this study. Results that were denoted as “n.a.” indicate that the algorithm

TABLE 1. Datasets parameters/properties

Data labels	F	H_f	i	N_i	Reference
fjs01	1	3	5	4	[4, 53, 54]
fjs02	1	10	100	n.a.	[53]
dfjs01a	2	3	10	4	[4, 56]
dfjs01b	2	3	10	4	[4, 56]
Mt06	1	6	6	6	[55]
Mt10	1	10	10	10	[55]
mt20	1	5	20	5	[55]

*a without maintenance integration, *b with maintenance integration
*n.a.: not available/no specific numbers of operation (flexible)

TABLE 2. MCRO control parameters

Parameter	fjs01,02	dfjs01a	dfjs01b	Mt06,10,20
Generation No.	500	100	5000	5000
Run No.	5	5	5	5
Collision Probabilities (p_{coll})	0.3	0.5	0.7	0.8
Options No.	4	4	4	4
Based on Option:	1	2	3	4
Population Size (pop_N)	5	10	15	30
Kinetic Energy Loss Rate (KE_{loss})	0.01	0.05	0.1	0.2

TABLE 3. Comparison of the results of the first and second datasets

Experiment 1									
Data Name	Ant Colony	*dev (%)*	GADG 1,2,3	*dev (%)*	MGADG	*dev (%)*	IGA	*dev (%)*	CRO
fjs01	42	+26.19	36	+13.89	35	+11.43	35	+11.43	**31**
fjs02	n.a.	n.a.	227	+0.88	n.a.	n.a.	n.a.	n.a.	**225**
dfjs01a	n.a.	n.a.	42	+16.67	n.a.	n.a.	37	+5.41	**35**
dfjs01b	n.a.	n.a.	122	+23.77	**93**	0.00	n.a.	n.a.	**93**

Experiment 2					
Data Name	MGA	*dev (%)*	IGA	*dev (%)*	CRO
Mt06	55	+12.73	55	+12.73	**48**
Mt10	972	+1.95	930	–2.47	953
Mt20	1207	+13.59	1172	+11.01	**1043**
Average Improvement		+9.42		+7.09	

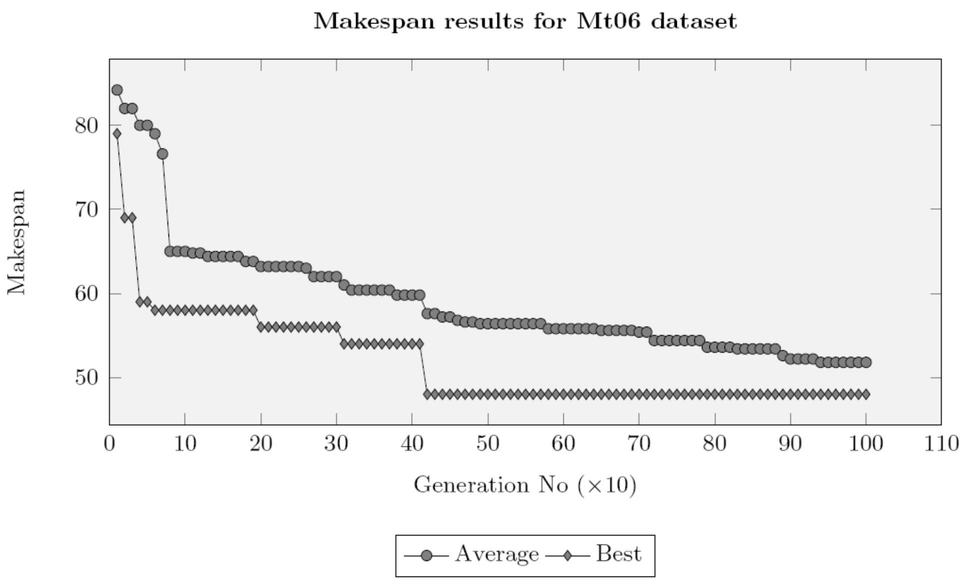

FIGURE 7. Makespan results for Mt06 dataset

consideration of the datasets is unavailable. The relative deviation obtained by MCRO compared with that of other algorithms for Experiment 1 are between $5\% \le dev \le 26\%$, whereas relative deviation for Experiment 2 is between $11\% \le dev \le 14\%$. In total, results obtained by IIA relatively deviate between $1\% \le dev \le 15\%$. Regardless of the run numbers, the optimal solution was achieved by MCRO compared with other algorithms. Thus, few test runs justify the capabilities of our proposed MCRO against other algorithms.

In terms of iteration sizes (generations), MCRO requires more iterations to converge compared with IGA because of the complexity of its operator in each iteration. Nevertheless, MCRO outperformed the other algorithms in the first test and in two out of three datasets in the second test (dataset Mt06 and Mt20). In addition, Figure 7 shows the decrease of the average makespan and the best makespan over five runs for the Mt06 dataset with 6 jobs and 6 machines. The figure indicates that our algorithms improved

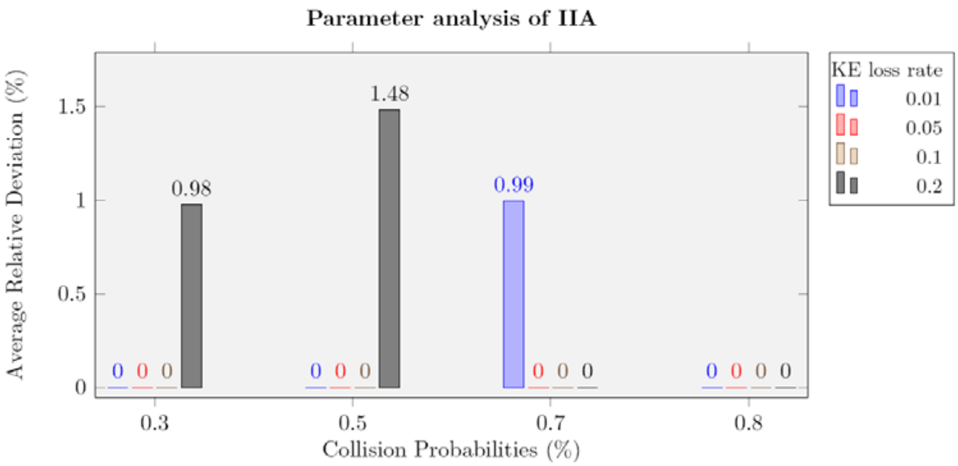

FIGURE 8. Parameter analysis

the average makespan very rapidly; the best global makespan (48) was achieved after 420 generations.

Regardless of the inclusion or exclusion of maintenance in FMSDS, MCRO still obtains superior results with the iteration numbers (generations) that of in Chan *et al.* [4, 53, 54]. MCRO also considers various combinations of parameters. Determining the appropriate parameters greatly influences the solutions and the probability of reducing premature convergence. As such, the appropriate parameter combinations were identified by analyzing different combinations of parameters. Specifically, the collision probabilities (p_{coll}) and KE loss rate (KE_{loss}) were investigated by capturing the value of average relative deviation (ARD) of the solution candidate. The detailed results of the different parameter combinations are graphically shown in Figure 8. The p_{coll} values are 0.3, 0.5, 0.7, and 0.8. The KE_{loss} values are 0.01, 0.05, 0.1, and 0.2.

6. **Result Analysis and Discussions.** From our observation, we could argue, with reference to Figure 8, that higher numbers of p_{coll} is better suited with higher numbers of KE_{loss} (e.g., p_{coll} of 0.7 or 0.8 that coupled with KE_{loss} of either 0.1 or 0.2). This is mainly because higher p_{coll} implies more collision happens in the molecular level which consequently resulted in higher loss of kinetic energy. As such, the potential solutions are less likely to deviate but only with small variation. On the other hand, we can observe two distinct situations with opposing values of p_{coll} and KE_{loss}. Firstly, "rapid" collision situation where high value of p_{coll} is coupled with low value of KE_{loss}. Secondly, "impacted" collision situation where low value of p_{coll} is coupled with high value of KE_{loss}. Both situations produce higher ARD of the solution candidate which implies superior diversity in the populations. However, both situations are uncertain in order to achieve global optimum or worse, trapped in local optimum. Nevertheless, any cases of p_{coll} and KE_{loss} would not guarantee the global optimum. As indicated in Figure 8, we can conclude that the combination of the lowest and highest KE_{loss} considered in this study with any p_{coll} produces an optimal result with small deviations. However, the best possible parameter combinations, as indicated in Figure 8, are the combinations of any p_{coll} with KE_{loss} of 0.05 and 0.1.

The problem in FMSDS subject to machine maintenance is considered as another alternative to reduce cost and to increase overall productivity. Such outcome is possible because a large number of machines can operate at an optimum level, and the possibility of machine breakdown can be reduced. Based on the results, we found that incorporating machine maintenance policy still maintains overall system performance, optimizes production scheduling plan, and reduces possible machine unavailability. Acknowledging the work of Jia *et al.* [7], we relish diversity and quick solution's evaluation because of the simplified encoding scheme in the proposed MCRO. From the results obtained by MCRO, optimum solutions are obtained with large iteration sizes, which are countered with fast computational time. Such feature indirectly promotes high productivity, equating significant resource usage. In addition, the solution obtained is relatively close compared with other algorithm's solutions applied in similar field. The greedy decoding scheme always guarantees a superior solution which consequently improves solution quality in each evaluation process. Therefore, MCRO is evidently suitable and competitive in solving the problem in FMSDS subject to machine maintenance. However, several gaps are identified in the proposed MCRO which are summarized as follows:

1. **Impacts of the parameters ($\boldsymbol{p_{coll}}$ and $\boldsymbol{KE_{loss}}$)**: CRO parameters are shown to have little impact on the solutions, whereas most meta-heuristic algorithms are largely dependent on parameters tuning in achieving good search results [57]. However, suitable parameter value, with respect to the problem domain, also plays roles in determining how much activity happens during CRO molecular processes which consequently produce much better results.
2. **Stability versus optimality**: Growing iterations numbers throughout the search progress had shown the ability of CRO to achieve optimum results. In addition, the considered size of the populations in this study which are considered small may also affect the results (less diverse). Larger population sizes would hypothetically reduce the iteration size needed, but potentially lead to high computational cost because of the dynamic population growth nature of the CRO (synthesis and decomposition molecular processes which decrease and increase population numbers, respectively).
3. **Computational complexity**: Due to intricate molecular processes involved in CRO, achieving optimum results requires careful emphasis on the detail of the molecule representation. More constraints and restrictions would impede possibility of diversifying the solution as well as achieving the global optimum. Otherwise, a different representation strategy to reduce the computational complexity of CRO is needed.

The MCRO has been proposed to solve the problem in FMSDS subject to machine maintenance. However, much considerable effort is required in order to merely solve a single aspect of real-world problems. Further improvements can be made on the following aspects with respect to the FMSDS problems:

1. Given the stochastic nature of MCRO, possible extension of MCRO with an artificial neural-network in order to find system-specific parameters or operating strategies to efficiently and effectively produce high quality solutions. Alternatively, possible integrated development with an expert system for obtaining scheduling knowledge in an FMSDS environment.
2. Rescheduling strategies can be incorporated in MCRO to improve solution quality in a real-time operation. Instead of focusing strategies on the algorithms, the strategies of handling the problems itself may provide fruitful results which consequently, enhance overall productivity.
3. MCRO can be coupled with an efficient machine maintenance strategy to improve solution reliability and quality. However, careful analysis on the matter should be emphasized because of probable increased computational complexity of the solution presentation.

4. Worst-case scenarios (i.e., machine breakdown) can be simulated to further test MCRO capabilities. Flexible and dynamic nature of the real production scheduling problems can further test the limits of MCRO in producing optimal solution. Furthermore, possible enhancements can also be identified.
5. A systematic methodology can be developed to add value to the major MCRO operators (e.g., collision, synthesis, and decomposition) specific to the scheduling problem.
6. Other hardware elements of the manufacturing system can be included to develop an integrated scheduling task. Therefore, MCRO can be applied in every element of the manufacturing system individually or in parallel, by conducting micro-level management of the elements of the manufacturing system using MCRO.

7. **Conclusion and Future Work.** This study proposed the MCRO algorithm to solve the problem in FMSDS subject to machine maintenance. The MCRO parameters and operators were presented, and comparisons with other algorithms in similar fields were conducted to justify the overall performance and optimization capabilities of the proposed MCRO algorithm. The satisfactory results obtained in this study serve as a motivation to extend this work to other complex and challenging environments of manufacturing, such as conforming to different technological requirements, considering cost reduction while inducing profit, managing skilled but scarce labor, and dynamically adapting to the costumer or consumer demands.

Nevertheless, datasets obtained from the literature and benchmarks are mere representations of real-word manufacturing problems, which are significantly more complex and difficult to comprehend. Achieving conceivable results that satisfy the actual manufacturing problem is still far from reality. Future work may include the extended or hybridized design of the MCRO algorithm, comparison of different maintenance strategies, large job and factory data, and consideration of comprehensive parameters and analytical results.

Acknowledgment. The main author wishes to thank Universiti Sains Malaysia for the support which has extended through grant number 304/PKOMP/6313026. The authors also gratefully acknowledge the comments and suggestions of the reviewers.

REFERENCES

[1] A. Bagheri, M. Zandieh, I. Mahdavi and M. Yazdani, An artificial immune algorithm for the flexible job-shop scheduling problem, *Future Generation Computer Systems*, vol.26, no.4, pp.533-541, 2010.
[2] F. T. S. Chan, H. Chan and H. Lau, The state of the art in simulation study on fms scheduling: A comprehensive survey, *The International Journal of Advanced Manufacturing Technology*, vol.19, no.11, pp.830-849, 2002.
[3] F. Chan, S. Chung and P. Chan, An adaptive genetic algorithm with dominated genes for distributed scheduling problems, *Expert Systems with Applications*, vol.29, no.2, pp.364-371, 2005.
[4] F. Chan, S. Chung, L. Chan, G. Finke and M. Tiwari, Solving distributed fms scheduling problems subject to maintenance: Genetic algorithms approach, *Robotics and Computer-Integrated Manufacturing*, vol.22, no.5, pp.493-504, 2006.
[5] W. Shen, Distributed manufacturing scheduling using intelligent agents, *IEEE Intelligent Systems*, vol.17, no.1, pp.88-94, 2002.
[6] M. Aboutalebi, H. Shirgahi and H. Motameni, Distributed flexible manufacturing system (FMS) scheduling using memetic algorithm, particle swarm optimization and timed petri net, *International Journal of the Physical Sciences*, vol.6, no.14, pp.3557-3563, 2011.
[7] H. Jia, A. Nee, J. Fuh and Y. Zhang, A modified genetic algorithm for distributed scheduling problems, *Journal of Intelligent Manufacturing*, vol.14, no.3-4, pp.351-362, 2003.
[8] L. De Giovanni and F. Pezzella, An improved genetic algorithm for the distributed and flexible job-shop scheduling problem, *European Journal of Operational Research*, vol.200, no.2, pp.395-408, 2010.

[9] M. Gholami and M. Zandieh, Integrating simulation and genetic algorithm to schedule a dynamic flexible job shop, *Journal of Intelligent Manufacturing*, vol.20, no.4, pp.481-498, 2009.
[10] D. Lee and F. DiCesare, Scheduling flexible manufacturing systems using petri nets and heuristic search, *IEEE Transactions on Robotics and Automation*, vol.10, no.2, pp.123-132, 1994.
[11] J. Paulli, A hierarchical approach for the fms scheduling problem, *European Journal of Operational Research*, vol.86, no.1, pp.32-42, 1995.
[12] A. Moro, H. Yu and G. Kelleher, Hybrid heuristic search for the scheduling of flexible manufacturing systems using petri nets, *IEEE Transactions on Robotics and Automation*, vol.18, no.2, pp.240-245, 2002.
[13] A. Reyes, H. Yu, G. Kelleher and S. Lloyd, Integrating petri nets and hybrid heuristic search for the scheduling of FMS, *Computers in Industry*, vol.47, no.1, pp.123-138, 2002.
[14] G. Mejía and N. Odrey, An approach using petri nets and improved heuristic search for manufacturing system scheduling, *Journal of Manufacturing Systems*, vol.24, no.2, pp.79-92, 2006.
[15] S. Wang, L. Xi and B. Zhou, Filtered-beam-search-based algorithm for dynamic rescheduling in FMS, *Robotics and Computer-Integrated Manufacturing*, vol.23, no.4, pp.457-468, 2007.
[16] S. Wang, L. Xi and B. Zhou, FBS-enhanced agent-based dynamic scheduling in FMS, *Engineering Applications of Artificial Intelligence*, vol.21, no.4, pp.644-657, 2008.
[17] L. Xing, Y. Chen and K. Yang, An efficient search method for multi-objective flexible job shop scheduling problems, *Journal of Intelligent Manufacturing*, vol.20, no.3, pp.283-293, 2009.
[18] P. Srinoi, E. Shayan and F. Ghotb, Scheduling of flexible manufacturing systems using fuzzy logic, *International Journal of Production Research*, vol.44, no.11, pp.1-21, 2002.
[19] K. Lee, Fuzzy rule generation for adaptive scheduling in a dynamic manufacturing environment, *Applied Soft Computing*, vol.8, no.4, pp.1295-1304, 2008.
[20] N. Shadbolt, Nature-inspired computing, *IEEE Intelligent Systems*, vol.19, no.1, pp.2-3, 2004.
[21] J. Holland, Genetic algorithms, *Scientific American*, vol.267, no.1, pp.66-72, 1992.
[22] S. Kirkpatrick, M. Vecchi et al., Optimization by simulated annealing, *Science*, vol.220, no.4598, pp.671-680, 1983.
[23] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, Equation of state calculations by fast computing machines, *The Journal of Chemical Physics*, vol.21, p.1087, 1953.
[24] W. Teekeng and A. Thammano, A combination of shuffled frog leaping and fuzzy logic for flexible job-shop scheduling problems, *Procedia Computer Science*, vol.6, pp.69-75, 2011.
[25] Y. Kim, J. Kim and K. Shin, An asymmetric multileveled symbiotic evolutionary algorithm for integrated fms scheduling, *Journal of Intelligent Manufacturing*, vol.18, no.6, pp.631-645, 2007.
[26] M. Dorigo and L. Gambardella, Ant colony system: A cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computation*, vol.1, no.1, pp.53-66, 1997.
[27] R. Eberhart and J. Kennedy, A new optimizer using particle swarm theory, *Proc. of the 6th International Symposium on Micro Machine and Human Science*, pp.39-43, 1995.
[28] L. De Castro and F. Von Zuben, Learning and optimization using the clonal selection principle, *IEEE Transactions on Evolutionary Computation*, vol.6, no.3, pp.239-251, 2002.
[29] D. Karaboga, An idea based on honey bee swarm for numerical optimization, *Techn. Rep.*, 2005.
[30] S. Rahmati and M. Zandieh, A new biogeography-based optimization (bbo) algorithm for the flexible job shop scheduling problem, *The International Journal of Advanced Manufacturing Technology*, vol.58, no.9, pp.1115-1129, 2012.
[31] S. Burnwal and S. Deb, Scheduling optimization of flexible manufacturing system using cuckoo search-based approach, *The International Journal of Advanced Manufacturing Technology*, vol.64, pp.951-959, 2012.
[32] Z. Geem, J. Kim and G. Loganathan, A new heuristic optimization algorithm: Harmony search, *Simulation*, vol.76, no.2, pp.60-68, 2001.
[33] F. Glover and C. McMillan, The general employee scheduling problem: An integration of MS and AI, *Computers & Operations Research*, vol.13, no.5, pp.563-573, 1986.
[34] N. Jawahar, P. Aravindan and S. Ponnambalam, A genetic algorithm for scheduling flexible manufacturing systems, *The International Journal of Advanced Manufacturing Technology*, vol.14, no.8, pp.588-607, 1998.
[35] J. Yang, Ga-based discrete dynamic programming approach for scheduling in FMS environments, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol.31, no.5, pp.824-835, 2001.

[36] T. Prabaharan, P. Nakkeeran and N. Jawahar, Sequencing and scheduling of job and tool in a flexible manufacturing cell, *The International Journal of Advanced Manufacturing Technology*, vol.29, no.7, pp.729-745, 2006.

[37] J. Gao, M. Gen, L. Sun and X. Zhao, A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems, *Computers & Industrial Engineering*, vol.53, no.1, pp.149-162, 2007.

[38] R. Kumar, M. Tiwari and R. Shankar, Scheduling of flexible manufacturing systems: An ant colony optimization approach, *Proc. of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol.217, no.10, pp.1443-1453, 2003.

[39] J. Jerald, P. Asokan, G. Prabaharan and R. Saravanan, Scheduling optimisation of flexible manufacturing systems using particle swarm optimisation algorithm, *The International Journal of Advanced Manufacturing Technology*, vol.25, no.9, pp.964-971, 2005.

[40] A. Prakash, F. Chan and S. Deshmukh, FMS scheduling with knowledge based genetic algorithm approach, *Expert Systems with Applications*, vol.38, no.4, pp.3161-3171, 2011.

[41] P. Udhayakumar and S. Kumanan, Sequencing and scheduling of job and tool in a flexible manufacturing system using ant colony optimization algorithm, *The International Journal of Advanced Manufacturing Technology*, vol.50, no.9, pp.1075-1084, 2010.

[42] H. Yang and Z. Wu, The application of adaptive genetic algorithms in FMS dynamic rescheduling, *International Journal of Computer Integrated Manufacturing*, vol.16, no.6, pp.382-397, 2003.

[43] Y. Shiue and R. Guh, Learning-based multi-pass adaptive scheduling for a dynamic manufacturing cell environment, *Robotics and Computer-Integrated Manufacturing*, vol.22, no.3, pp.203-216, 2006.

[44] A. Y. Lam and V. O. Li, Chemical-reaction-inspired metaheuristic for optimization, *IEEE Transactions on Evolutionary Computation*, vol.14, no.3, pp.381-399, 2010.

[45] J. Sun, Y. Wang, J. Li and K. Gao, Hybrid algorithm based on chemical reaction optimization and lin-kernighan local search for the traveling salesman problem, *The 7th International Conference on Natural Computation*, vol.3, pp.1518-1521, 2011.

[46] T. K. Truong, K. Li and Y. Xu, Chemical reaction optimization with greedy strategy for the 0-1 knapsack problem, *Applied Soft Computing*, vol.13, no.4, pp.1774-1780, 2013.

[47] B. Alatas, ACROA: Artificial chemical reaction optimization algorithm for global optimization, *Expert Systems with Applications*, vol.38, no.10, pp.13170-13180, 2011.

[48] S. Yang, Y. Yi and Z. Shan, Gbest-guided artificial chemical reaction algorithm for global numerical optimization, *Procedia Engineering*, vol.24, pp.197-201, 2011.

[49] J. Xu, A. Y. Lam and V. O. Li, Chemical reaction optimization for the grid scheduling problem, *IEEE International Conference on Communications*, pp.1-5, 2010.

[50] J. Xu, A. Y. S. Lam and V. O. K. Li, Chemical reaction optimization for task scheduling in grid computing, *IEEE Transactions on Parallel and Distributed Systems*, vol.22, no.10, pp.1624-1631, 2011.

[51] J.-Q. Li and Q.-K. Pan, Chemical-reaction optimization for flexible job-shop scheduling problems with maintenance activity, *Applied Soft Computing*, vol.12, no.9, pp.2896-2912, 2012.

[52] E. Rodriguez-Tello, J.-K. Hao and J. Torres-Jimenez, A comparison of Memetic recombination operators for the MinLA problem, *MICAI: Advances in Artificial Intelligence*, pp.613-622, 2005.

[53] F. Chan, S. Chung and P. Chan, Application of genetic algorithms with dominant genes in a distributed scheduling problem in flexible manufacturing systems, *International Journal of Production Research*, vol.44, no.3, pp.523-543, 2006.

[54] F. Chan, S. Chung and L. Chan, An introduction of dominant genes in genetic algorithm for FMS, *International Journal of Production Research*, vol.46, no.16, pp.4369-4389, 2008.

[55] H. Fisher and G. L. Thompson, Probabilistic learning combinations of local job-shop scheduling rules, *Industrial Scheduling*, pp.225-251, 1963.

[56] S. H. Chung, F. T. S. Chan and H. K. Chan, A modified genetic algorithm approach for scheduling of perfect maintenance in distributed production scheduling, *Engineering Applications of Artificial Intelligence*, vol.22, pp.1005-1014, 2009.

[57] E.-G. Talbi, Common concepts for metaheuristics, *Metaheuristics: From Design to Implementation*, pp.1-86, 2009.