Improved Immune Algorithm for Optimizing Distributed Production Scheduling Problem in Flexible Manufacturing System Subject to Machine Maintenance

Mohd Nor Akmal Khalid[1], Umi Kalsom Yusof[1]
[1]School of Computer Sciences,
Universiti Sains Malaysia,
11800 USM, Penang, Malaysia
mnak104041@student.usm.my, umiyusof@cs.usm.my

Abstract—Competitiveness and rapid expansion of flexible manufacturing system (FMS), as one of the industrial alternatives, has attracted the attention of many practitioners and academicians. Globalization has further encouraged FMS development into distributed, self-reliant units of production center. Flexible manufacturing system in distributed system (FMSDS) considers multi-factory environments, where jobs are processed by a system of FMSs. Scheduling problems in FMSDS deal with allocation of jobs to factories, independent assignment of job operation to machines, and operations sequencing on a machine. In addition, many previous studies neglect the impact of maintenance as one of the core parts of production scheduling. Maintenance significantly affects the overall performance of production scheduling. As such, maintenance is considered in this paper as part of production scheduling. This paper aims to minimize the global makespan over all the factories. This paper proposes an improved immune algorithm (IIA) to solve scheduling problems in FMSDS subjects to machine maintenance. Antibody encoding is adopted to explicitly represents information about a factory, job, and maintenance, while a greedy decoding procedure exploits scheduling flexibility and determines the job routings. Instead of the traditional mutation operator, an improvised mutation operator is used to improve the solutions by refining the most promising individuals within passing generation (iteration). The proposed approach was compared with an ant colony algorithm and several variants of genetic algorithms. The proposed IIA improved the global makespan from 4% to 33% compared with other algorithms. IIA performance has also been tested with several adjustments of population sizes, clonal selection rates, and local explorative mutation rates.

Keywords—distributed manufacturing, flexible manufacturing system, artificial immune system, preventive maintenance, optimization

I. Introduction

PRODUCTION scheduling problems in the manufacturing industry have been the subject of research initiatives for several years. Developments in computer technology have promoted new ways of solving problems related to production scheduling. The newly developed methods have rendered exact approaches insufficient to handle complex and changing environments of production scheduling. In general, production scheduling involves allocating a limited amount of resources (i.e., machine) to a number of tasks over a time horizon. With the many available feasible solutions for different task-resource assignments, production scheduling problems are considered one of the *NP*-hard problems [1]. Given this issue, both practitioners and academicians aim to solve production scheduling problems.

Strong market competition and challenging manufacturing environment a have necessitated the evolution of the ways organizations attain success and gain a competitive edge. Flexible manufacturing system (FMS) is the result of growing demand for quantity and quality. FMS combines the efficiency of a high-production line and the flexibility of job shops, which makes it suitable for mid-volume batch production and mid-variety of products [2]. FMS offers high investment value and benefits, which gives it a competitive advantage in today's challenging and globalized market. Such benefits can be exploited to achieve the best efficiency in this particular area. FMS has been studied extensively since the 1980s, and most studies focus on allocation, scheduling, loading, and control problem. The scheduling problem in FMS can be further defined as an optimization of one or more system criteria with consideration of a limited set of resources while taking various constraints into account.

Production scheduling of a single-factory is directed toward minimization of total operating cost, completion time, and order fulfillment of assigned machine to process the job's operations. Today's trend of globalization has cultivated the emergence of the distributed scheduling (DS) in FMS. DS can be defined as a multi-factory production system where each factory is geographically distributed and possesses the ability

Manuscript received November 19, 2013; revised January 11, 2014.

to process product parts independently. Consequently, DS may yields different production lead times, operating costs, and completion times [3], [4]. Thus, exact and heuristic solutions becomes insufficient and unscalable, which makes them unable to handle scheduling problems in flexible manufacturing system distributed scheduling (FMSDS) that considers different combinations of process plans. As such, scheduling in FMSDS has been considered directly in recent works, including [3], [5], [6], [7].

Thus, with respect to available studies intended to solve scheduling problems in FMSDS, production schedule optimization can be concluded to involves three hierarchical problems that need to be solved sequentially or simultaneously [8], [9]:

1) Allocation of the most suitable factory for the job (assignment problems).
2) Routing of the most suitable machine for each 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 reasons for machine unavailability for a period of time [10]. Research efforts on machine maintenance have increased because machine maintenance directly affects the production rate, product quality, machine availability and utilization ratio [4]. The lack of machine maintenance disrupts predetermines plan or scheduling as a result of process mismatching during machine unavailability. Thus, maintenance policy in production scheduling is vitals in ensuring machine availability and utilization ratio, while maximizing the facility with minimum cost and reducing unforeseen breakdowns. To the best of our knowledge, Chan et al. [4], who proposed genetic algorithm with dominated genes (GADG), is the first work that addresses all the features of FMSDS and considers machine maintenance.

Solutions to optimization problems are inspired by several mechanisms that form the building block of a very complex natural immune systems defense against pathogenic organisms. The immune algorithm (IA) is a meta-heuristic which is developed based on such system. Explorative powers of hypermutation operator, solution diversity through its receptor editing operator, and evolutionary capacity of memory cell in IA makes this algorithm a suitable alternative in the optimization domain. As such, this paper proposes the use of AI to solve scheduling problems in FMSDS subjects to machine maintenance. The objectives of this study include proposing an improved IA with guided initialization mechanism and yielding optimal or lowest makespan while considering the impact of machine maintenance.

The remainder of this paper is organized as follows: Section II discusses previous works that were conducted to solve scheduling problems in FMS. Section III highlights the problem statement, constraints and parameters of distributed production scheduling of an FMS subject to maintenance. Section IV discusses the proposed solution in detail. Section V elaborates the results of the proposed approach, compares the parameters, and compares the proposed approach with other algorithms. Section VI concludes this paper.

II. Related Works

Different methods have been proposed to solve scheduling problems in FMS. Lee and DiCesare [11] formulated a Petri net model for a single manufacturing environment. The model was used to model the routing flexibility, resource sharing, lot sizes, and concurrency of a production schedule. Paulli [12] proposed a hierarchical algorithm based on similarities with the job-shop scheduling problem in FMS. Reyes *et al.* [13] further enhanced the Petri net model with a hybrid search algorithm to mitigate the complexity of the problem. Kumar *et al.* [14] introduced a different approach that used an ant colony optimization (ACO) method that applied a graph-based representation where a collective outcome of all solution found by the ant will be the final solution of the algorithm. Jerald *et al.* [15] tested genetic algorithm (GA), simulated annealing, memetic algorithms, and particle swarm optimization algorithms. Different problem complexities were conducted on each approach to test their performances. In all cases of the problems, particle swarm optimization produced the best result compared with the others. Wadhwa *et al.* [16] designed a knowledge-based GA that is integrated with the classical genetic algorithm to generate initial population, selection operator, and crossover operator. Burnwal and Deb [17] used cuckoo search-based algorithm to optimize scheduling problems in FMS, which is compared with the performance of the GA and the particle swarm algorithm.

Shen [5] proposed a hybrid of agent-based algorithm and genetic algorithm in the distributed scheduling system. Features such as encapsulation, coordination and negotiation, and decision schemes between agent-based algorithm and GA were studied to solve scheduling problems in FMSDS. An adaptive genetic algorithm with dominant genes was introduced by Chan *et al.* [3] to solve scheduling problem in FMSDS. Chan *et al.* [4], [18] also enhanced genetic search performance by using dominant genes, modified crossover mechanism, and saturation operator for monitoring the similarity of the solution pool. Jia *et al.* [6] introduced a hybrid of genetic algorithm and Gantt chart to derive the schedules. This study achieved minimum makespan, job tardiness, or manufacturing cost of small-sized and medium-sized scheduling problem in FMSDS. Aboutalebi *et al.* [7] introduced a combination of memetic algorithm, particle swarm optimization and timed Petri net algorithm to solve

scheduling problems in FMSDS. In the proposed method, the FMSDS is formulated by timed Petri net. Scheduled tasks are conducted by memetic algorithm and particle swarm optimization methods in which reasonable results are obtained and compared with available approaches in the literature.

Preventive maintenance (PM) scheduling problems in other problem domains have been explored by previous researchers. Gopalakrishnan *et al.* [19] developed a tabu search algorithm with the aim to maximize priority tasks subject to resource availability constraints while performing tests with a benchmark problem that successfully reduced the optimality gap. Particle swarm optimization is proposed by Pereire *et al.* [20], who focused on reliability and cost evaluation by using a probabilistic model. PM in FMS equipment was studied by Xue *et al.* [21], who proposed ant colony optimization (ACO) based on multiple ant colonies. The ant colonies were divided into general-ant colony and core-ant colony. The effectiveness and the efficiency of the adaptive strategies applied in the algorithm were demonstrated through the reasonable results obtained.

Few studies discuss PM for scheduling problems in FMSDS. A genetic algorithm with dominant genes was introduced by Chan *et al.* [4], who later enhanced the algorithm with premature avoidance and local search strategy [18]. Three separate studies were conducted by Chung *et al.* [22] to prove the influences of PM for scheduling problems in FMSDS by using the same approach as in [3], [4], [18]. Different maintenance models were tested, and the results were compared with prior research. However, few studies focus on solving FMSDS-related problems by using IA.

IA, which is based on the adaptive natural immune system of vertebra, has been used in various applications, such as machine learning, pattern recognition and detection, scheduling, intrusion detection, data manipulation and analysis, evolutionary computation, and optimization [23], [24], [25], [26], [27], [28], [29]. IA features that are not limited to self-organizing, adaptivity, and uniqueness have the potential to be used in developing computational models applied in business [23], [25], [29], [30], [31], sciences and engineering [27], [28], [32], [33], and optimization domain [24], [33], [34], [35]. Generally, IA is known for its advantages, such as memory cells (reservation of good solutions), high-rate of somatic mutation or hyper-mutation (explorative and/or diversification mechanism), and receptor editing (escaping local optima, adaptivity) [36]. Despite available studies on IA, this research was conducted in response to an encouraging yet challenging opportunity to use a renowned IA as a suitable choice to address underlying problems in FMSDS subject to machine maintenance.

III. Problem Description

The notations used to describe the problem studied throughout the paper are given in Table I.

TABLE I
PROBLEM NOTATION

Notation	Description
f	index for factory, f = 1, , F, where F is the number of factories
i	index for job, i = 1, , I, where I is the number of job
j	index for operation, j = 1, , N_i, where N_i is the number of operation in job i
h	index for machine, f = 1, , H_f, where H_f is the number of machine in factory f
k	index for time slot, k = 1, , K, where K is the maximum time horizon
D_{if}	The delivery time required to deliver product from the location of factory f to the location of job i
T_{ijfh}	operating lead time of operation j of job i on machine h in factory f
M	maximum machine age
S_{ij}	starting time of operation j of job i
E_{ij}	ending time of operation j of job i
C_i	completion time of job i
χ_{if}	= 1, if job i is allocated to factory f = 0, otherwise
δ_{ijfhk}	= 1, if operation j of job i occupies time slot k on machine h in factory f = 0, otherwise
γ_{ijfh}	= 1, if machine h in factory f is maintained after operation j of job i = 0, otherwise

The FMSDS problem 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, ..., F) will be assigned to the job to generate corresponding production scheduling. Each individual factory has a number of machines (h = 1, 2, ..., H_f) with different 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 by more than one machine (not all), but must be in the same factory. The traveling time between factory f and job i is denoted as D_{if}.

A number of jobs (i) are expected to be received in the distributed network, and a suitable factory (f = 1, ..., F) will be assigned to the job to generate corresponding production scheduling. Each individual factory has a number of machines (h = 1, 2, ..., H_f) with different efficiencies or operating lead times (T_{ijfh}) in producing various product types. Each job has up to N_i numbers of operations, and every operation can be performed by more than one machine (not all), but in the same factory only. The traveling time between the factory f and the job i is denoted as D_{if}.

Each machine conforms to a maximum machine age (M), where the machine age equals the cumulated processing time of operations. A maintenance procedure has to be carried out

right after the completion of the current operation when the machine age reaches the threshold denoted as M, outlined in [4]. After every maintenance procedure, the machine age of the particular machine will be reset to 0, as shown in Fig. 1.

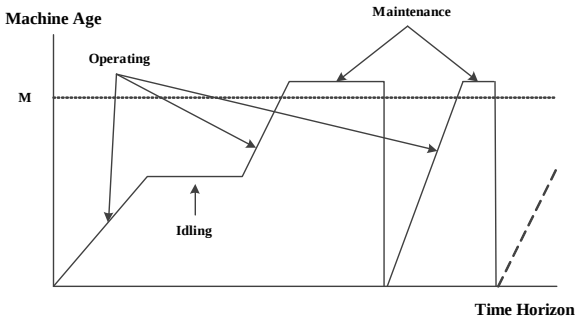

Fig. 1. Sample of machine age modeling for a machine when reaching maximum machine age adopted from [22].

The maintenance model used in this paper is a linear relationship between the required maintenance time and the machine age, where maintenance time equals three times the machine age. For example, if the machine age is 40 units of time, then the maintenance time is 120 units of time. This maintenance model will simultaneously affect the results of the FMS distributed production scheduling. The maintenance model is shown in Fig. 2.

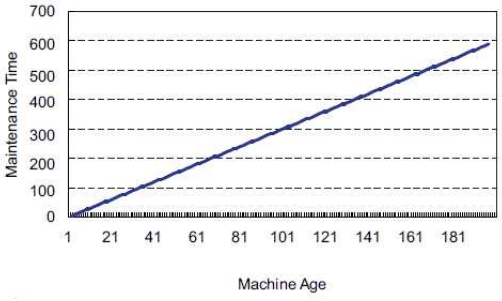

Fig. 2. Relationship between maintenance time and machine age for linear relationship model adopted from [22].

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

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

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

The problem is subject to the following constraints:
Precedence constraints:

$$S_{ij} \geq E_{i(j-1)} \quad (i = 1, 2, ..., I; j = 2, 3, ..., N_i). \tag{3}$$

Processing time constraints:

$$E_{ij} - S_{ij} = \sum_{fh} \chi_{if} T_{ijfh} \quad (i = 1, 2, ..., I; j = 1, 2, ..., N_i). \tag{4}$$

$$\sum_{fhk} \delta_{ijfhk} = \sum_{fh} \chi_{if} T_{ijfh} \quad (i = 1, 2, ..., I; j = 1, 2, ..., N_i). \tag{5}$$

Operation constraints:

$$\sum_{fhk} \delta_{ijfhk} = 1 \quad (i = 1, 2, ..., I; j = 1, 2, ..., N_i). \tag{6}$$

Processing operation constraints:

$$\sum_{fh} \delta_{ijfhk} \leq 1 \quad (i = 1, 2, ..., I; j = 1, 2, ..., N_i; k = 1, 2, ..., K). \tag{7}$$

Machine capacity constraints:

$$\sum_{ij} \delta_{ijfhk} \leq 1 \quad (k = 1, 2, ..., K; h = 1, 2, ..., H_f; f = 1, 2, ..., F). \tag{8}$$

Factory constraints:

$$\sum_{f} \chi_{if} = 1 \quad (i = 1, 2, ..., I). \tag{9}$$

According to Constraint 3, as given above, every operation can only begin after the completion of the prior operation. Constraint 4 states that when an operation commences, it will continue until it finishes without any disruption. Constraint 5 states that the assigned time slot must be equal to the required operation time. Constraint 6 requires each operation to be carried out on a single machine throughout the horizon. Constraint 7 requires each operation to be executed on a single machine at each unit of time. Constraint 8 requires each machine to handle only a single operation at each unit of time. Constraint 9 indicates that each job can only be assigned to a single factory.

IV. An IA for Solving Scheduling Problems in FMSDS Subjects to Maintenances

To further explain how IA can be implemented for optimization domain, the following section will elaborate on the generic IA with its standard operators. The proposed improved IA is then presented. The encoding and decoding process of the solution, solution initialization and clonal selection, and hyper-mutation are discussed to clarify the proposed improved IA differs from the generic IA. Hyper-mutation is also further divided into two consecutive sections, namely, local and global mutations.

A. Generic IA

IA is a collection of complex adaptive pattern recognition systems that mimic the natural immune system that defends an organism from foreign antigens (bacteria or viruses) by detecting, identifying, and killing pathogens and tumor cells (antigens) to protect it against diseases [24]. The system can recognize or identify cells (or molecules) within the organism as either harmful (non-self-cell) or harmless (self-cell) [1] to allow the system to naturally evolve to recognize and neutralize threats. In a typical infection process, infestation and proliferation of a pathogen within the organism occurs. Pathogens and antigens correspond to specific foreign proteins.

When a harmful non-self-cell enters the body, the immune system responds by providing immediate but nonspecific defense to protect the organism from any possibilities of an infection (innate immunity) [24]. Phagocyte, which is an antigen-presenting cell, will detect the presence of non-self cells and fight them by secreting T-cell-activating molecules. If this level of the immune system is penetrated by antigens or pathogens, the system initiates adaptive behavior (adaptive immunity) [1]. The activated T-cells select appropriate B-cells, which have receptors that closely resemble the antigenic or pathogenic signatures of the foreign proteins (clonal selection hypothesis). Then, these B-cells attach to the detected foreign protein's signature (binding site/epitope); this attachment process is known as affinity. Affinity is the measure for evaluating the successful binding of foreign proteins and B-cells [26]. This scenario is illustrated in Fig. 3.

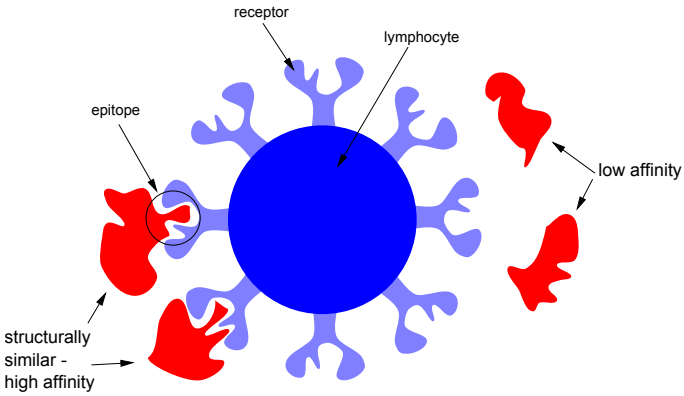

Fig. 3. Illustration of B-cells attach themselves to the detected foreign protein's signature (binding site/epitope) adopted from [37].

The B-cells then modify themselves (cellular reproduction) via somatic hyper-mutation or receptor editing to attain better affinity against antigens or pathogens by rapidly mutating or randomly changing their receptors' genetic orientation, respectively (affinity maturation). Afterwards, B-cells start to proliferate to produce clones, with a large number of identical B-cells being duplicated. Some of the mature B-cells will produce new plasma cells while others with high affinity threshold will be sustained as long-lasting memory cells [1]. The plasma cells secrete a large number of antibodies, which are distributed randomly throughout the blood and lymph systems, and are capable of recognizing and killing foreign proteins as well as detecting and recognizing malfunctioning self-cells. The memory B-cells with long life spans remain in the system to effectively accelerate the response of the immune system to future exposure to similar infection, while the other remaining clones of B-cells will die or be replaced by another new clone. The overall summary of generic IA discussed so far is shown in Fig. 4.

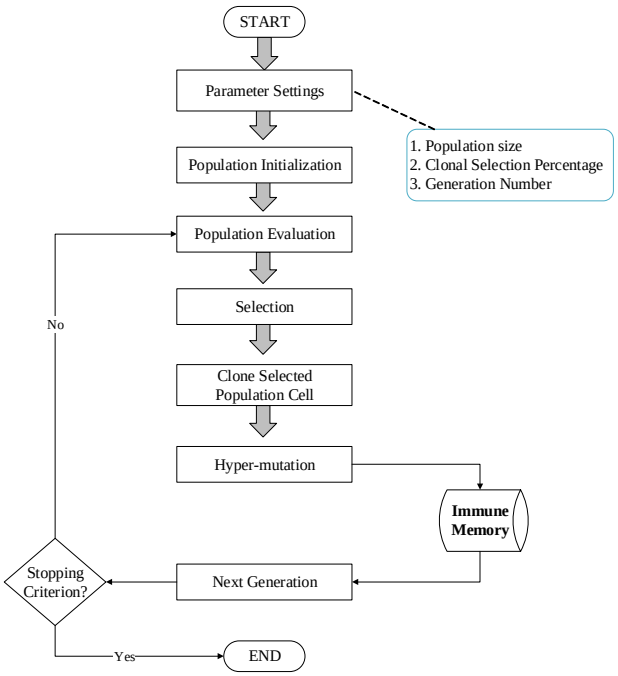

Fig. 4. Flowchart of generic IA.

B. Proposed Improved Immune Algorithm (IIA)

To implement the mechanisms explained in Section IV-A to address scheduling problems in FMSDS, several adaptations and adjustments are needed. The original IA must metaphorically match the imposed scheduling problems in FMSDS to meet the objective. For clarity, the analogy of IIA for this particular study is given in Table II.

The overall flows of the proposed IIA are shown in Fig. 5 to illustrate the proposed IIA procedures. The improved version of IA operators is highlighted by the thick border in the figure. The first procedure involves setting the parameters, where user-defined parameters, such as population size (pop_N), generation number, and clonal selection percentage (C_r), are assigned an individual value. The first step is then followed by the initialization of the populations (see Section IV-B2 for details), population ranking, and clonal selection. Simple

TABLE II
IIA ANALOGY

Immunology Terms	Scheduling Terms
B-cells	Machine available
Antibody	Machine assignment
Foreign protein	Job allocation
Receptor	Assigned job-machine
Affinity	Evaluation measures (makespan)
Affinity maturation	Improvisation process
Cloning	Duplication
Hyper-mutate	Improvisation scheme #1
Receptor editing	Improvisation scheme #2
Binding site/Epitope	Job assigned to a machine
High Affinity	Job assigned to a machine (with low makespan)

encoding during the initialization phase and greedy-based decoding scheme (see Section IV-B1 for details) during evaluation phase are conducted. After the clonal selection phase, a set of individuals are selected from the total population size where cloning is performed first, followed by somatic mutation (hyper-mutation) conducted on the cloned individual. At this point, only the local mutation operators are involved. Next, receptor editing (global mutation) is performed on one or more individuals of the population based on a probabilistic scheme (see Section IV-B3 for details on both local and global mutations). Then, the best among cloned individuals will be retained as an immune memory for the remaining generation numbers (iterations). If the termination condition (maximum generation number) is met, the proposed IIA terminates.

1) Antibody Encoding and Decoding: Information encoded in the antibody of the IIA for FMSDS has to specify the allocation of each job to factory, the routing of every job through machine, and the sequence of the operations. Basically, this study reuses the benefit of simple operation-based encoding method proposed in [8] for the distributed scheduling problems without routing flexibility, where relevant extension that includes flexibility issues of the FMSDS is considered. The receptor size (r_p) within an antibody is equal to the total number of operations of all jobs. Every receptor is represented by a triplet notation (*f, i, p*), which denotes the factory (*f*), the assigned job (*i*), and the PM flag (*p*). Note that all the operations of the same job are represented by different receptors within the same antibody, which are interpreted according to the order of receptor occurrence on the antibody given that the order for the operation of a job is fixed. Concerning the adoption of the simple representation as per [8], no information about alternative machine routes is explicitly encoded into the receptor. This information is retrieved during the decoding phase. A sample individual is given in Fig. 6.

Job 1, job 2, and job 3 are assumed have two, two, and three operations, respectively, so that an antibody consists of 7

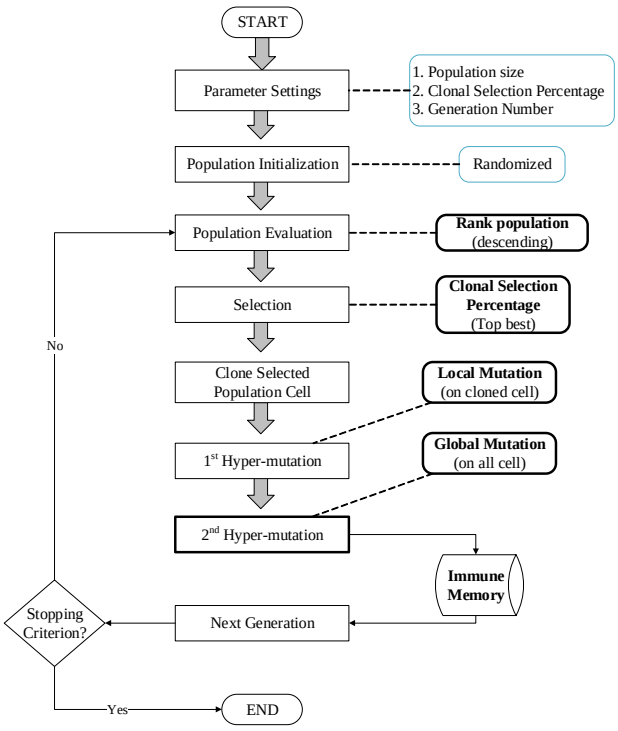

Fig. 5. Flowchart of proposed IIA

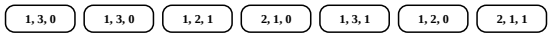

Fig. 6. Sample antibody encoding

receptors. Each receptor consists of three types: "2,1,$< p >$", "1,2,$< p >$", and "1,3,$< p >$" which means that jobs N_1 and N_2 are processed in factory F_1 and N_3 is processed in factory F_2. The decoding process exploits the provided information by each antibody to generate a schedule, and the affinity of each antibody is evaluated afterwards. The objective of the scheduling problems in FMSDS is to minimize the global makespan of the factory network to ensure that the affinity of an antibody is inversely related to the global makespan.

Antibodies explicitly represent information of job assignments to factories, and the order of the antibody's receptor is relevant to determine the priority of each operation, with no information on job routing considered. Instead of complicating gene encoding, the flexibility problem is considered in the decoding phase, where it can dispatch job operations to one of the alternative machines within the selected factory. Thus, information on job routing is conducted implicitly within the decoding process. Based on the order determined by the antibody's receptors, 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 always guarantees the lowest current local makespan where the one that provides the lowest completion time for the operations assigned so far is selected. However, if different routings lead to the same current local makespan, the machine with the smallest processing time is chosen. If the available machines have the same smallest current local makespan and processing time, any of them is selected at random in order 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 the job is assigned to) as soon as all the operations have been scheduled, thus obtaining the local and global makespans.

2) Population Initialization and Clonal Selection: The initial population is determined in three phases: the first phase randomly generates jobs until all the operations of the jobs are generated; the second phase randomly assigns jobs to factories in which related operations of the respective jobs will be amended to satisfy factory allocation constraints; and the third phase generates the maintenance flag at random. This process repeats until all individuals of the population (pop_N) are initialized. Note that these three phases are conducted with respect to the encoding and decoding schemes in the previous section.

During the clonal selection phase, a set of individuals from the current population are chosen to apply IIA operators and generate high affinity memory cell(s) to include in the next generation. The clonal selection is dependent on the affinity (makespan) of the antibody. Therefore, a ranking strategy is conducted by sorting the population by decreasing affinity (starting from the best to the worst individual). The C_r % of the best population will be considered for cloning, in which each of these cloned cells undergo affinity maturation process, whereas the rest of the population will be re-initialized with the three-phase initialization mechanism previously described. This re-initialization is conducted to give good antibodies more chances for affinity maturation, while giving even less promising individuals the opportunity to participate in the evolution; the search will move toward most promising regions while guaranteeing a certain diversity of the solution pool and preventing premature convergence of the algorithm.

3) Hyper mutation Operators: In this study, the mutation operators shares coherent behaviours with GA, where both have a mutation operator that either randomly generates a string or a decimal, or randomly flips a binary digit of the individual. However, IIA mutation operator differs in that, depending on individual affinity, inferior antibodies mutate at a higher rate compared with superior antibodies. This process is known as somatic mutation (hyper-mutation). To comply with this requirement, the mutation operator is conducted in a continuous loop in which the loop limit is calculated as follows:

$$R_m = Round\{(1 - A_{pop_n}) * pop_N\} \quad (10)$$

where R_m is the mutation rate, A_{pop_n} is the affinity of the nth population, and pop_N is the total population size. The somatic mutation used here can be categorized into local and global mutations.

a) Local Mutation: Local mutation is involved in exchanging information of the antibody's receptor, which is conducted on a single antibody where only routing of the operations of jobs are affected (local effects). Local mutation aims to enhance the algorithms to better examine the search space. Two types of local mutations operator are employed, namely, **uniform** and **exploration**. The uniform mutation operator is conducted repeatedly when the mutation is in the somatic mutation loop, whereas the exploration mutation operator mutates based on user-defined probabilities within the somatic mutation loop.

Simple swapping mechanism (SSM) is a uniform local mutation operator that randomly selects a pre-defined number of pair of receptors within a single antibody to permute their positions (Fig. 7). However, an end-to-end swapping mechanism (EESM) is employed as an exploration local mutation operator that exchanges first and last pairs of every receptors within a single antibody to permute their positions (Fig. 8). Note that every antibody explicitly encodes only the job information. However, exchanging the antibody's receptor does not effect the feasibility of scheduled job routing.

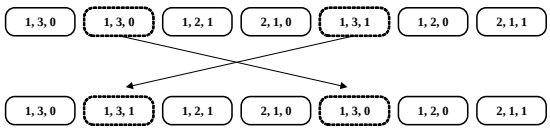

Fig. 7. Illustration of SSM.

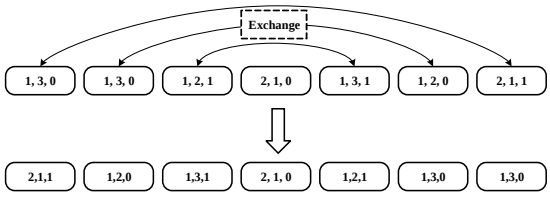

Fig. 8. Illustration of EESM.

b) Global Mutation: Global mutation involves exchanging information of the antibody's receptor, which is conducted on a single antibody at a time and involves factory assignment of jobs and the maintenance flag. Global mutation aims to explore more solutions of the search space with different assignments of jobs to factories and varied scheduled maintenance. To maintain consistency with the antibody's remaining receptors and meet factory constraints, all receptors have to

reflect the new job assignments in which all the receptors related to the selected job in the antibody have to be updated (global effects). The updating process is conducted on all antibodies after the last immune operator to maintain the antibody's feasibility (receptor editing).

Two types of global mutation are considered (Fig. 9): random factory assignment (RFA) and random scheduled maintenance (RSM). Global mutation has a significant effect on operation scheduling, which is why it is applied at some iterations based on certain probabilities to allow the algorithm to explore solutions to a given job assignment before changing it. As such, two additional parameters are defined based on Equations 11 and 12, respectively: the probability of applying RFA mutation (R_{g1}) and the probability of applying RSM mutation (R_{g2}), both applied to every generation.

$$R_{g1} = (1 - R_m)/2 \tag{11}$$

$$R_{g2} = (1 - R_m)/3 \tag{12}$$

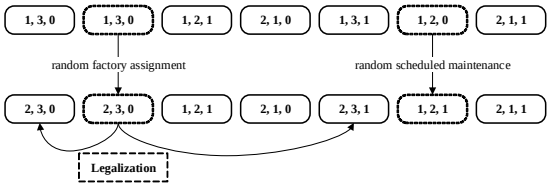

Fig. 9. Global mutation: RFA and RSM.

where R_{g1} is the mutation rate for RFA, R_{g2} is the mutation rate for RSM, and R_m is the mutation rate. As mentioned previously, global mutation significantly affects operation scheduling. Whenever a local mutation is not performed, global mutation is conducted with a reduced probability. For RFA, the probability is reduced in half ($\frac{1}{2}$), while the RSM probability is reduced by one-third ($\frac{1}{3}$). As such, the global mutation will be applied to some parts of the population's individual. Note that R_m mutation rate is used because the global mutation is directly proportional to the individual's affinity. The use of this mutation rate minimally adjusts the available population's individual and provides diversity within the overall population.

V. Computational Results and Discussion

The performance of the IIA has been tested in several instances. Two separate experiments were conducted. The first experiments was based on dataset that was obtained from Chan *et al.* [4], [38], [39], while the second experiment was based on datasets that were obtained from Fisher and Thompson's benchmark data [40]. The first experiment compares IIA with other algorithms designed for FMSDS (with maintenance for a specific dataset), in particular, ACO by Kumar *et al.* [14], GA with Dominant Gene (GADG) by Chan *et al.* [4], [38], [39], modified GADG (MGADG) by Chung *et al.* [41], and improved GA (IGA) by De Giovanni and Pezella [9]. The second experiment compares IIA with other algorithms that were used on the benchmark dataset; these algorithms are modified GA (MGA) by Jia *et al.* [8] and IGA by De Giovanni and Pezella [9]. IIA was implemented in C# compiler and run independently on a personal computer equipped with a 2.0 GHz Intel Core i5 processor and 2 GB RAM.

All datasets considered in this study are summarized in Table III. Both experiments were run independently and the best results among five test runs were recorded. IIA parameters were calibrated for the preliminary test on all datasets described above. The details of four parameters for each dataset considered are given in Table IV.

TABLE III
Datasets parameters/properties

Experiment 1					
Data labels	F	H_f	i	N_i	Reference
fjs01	1	3	5	4	[4], [38], [39]
fjs02	1	10	100	n.a.	[38]
dfjs01a	2	3	10	4	[4], [41]
dfjs01b	2	3	10	4	[4], [41]
Experiment 2					
Data labels	F	H_f	i	N_i	Reference
Mt06	1	6	6	6	[40]
Mt10	1	10	10	10	[40]
Mt20	1	5	20	5	[40]

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

TABLE IV
IIA control parameters

Parameter	fjs01,02	dfjs01(a)	dfjs01(b)	Mt06,10,20
Generation No.	500	100	5000	5000
Run No.	5	5	5	5
Local Explorative Mutation Rate (R_{eesm})	0.05	0.1	0.15	0.3
Options No.	4	4	4	4
Based on Option:	1	2	3	4
Population Size (pop_N)	50	75	150	300
Clonal Selection Rate (C_r)	0.25	0.45	0.65	0.75

Results of the first and second experiments are given in Table V. 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 IIA. The relative deviation is defined as in 13.

$$dev = [(MK_{comp} - MK_{IIA})/MK_{comp}] * 100\% \tag{13}$$

MK_{IIA} is the makespan obtained by the proposed IIA, and MK_{comp} is the other algorithm that was presented for comparison. As given in Table V, IIA outperforms other algorithms by obtaining optimal results for every datasets on both experiments considered in this study. Results that were denoted as "n.a" indicate that the algorithm consideration of the datasets is unavailable. The relative deviation obtained by IIA compared with that of other algorithms for Experiment 1 are between $5\% \leq dev \leq 33\%$, whereas average relative deviation for Experiment 2 is between $11\% \leq dev \leq 14\%$. In total, results obtained by IIA relatively deviate between $4\% \leq dev \leq 33\%$. Although five runs seems few, IIA shares relatively coherent convergence rate with GA (e.g., [42]), which generally requires a higher number of generation numbers. However, the optimal solution was achieved compared with other algorithms. Thus, few test runs can support the capabilities of our proposed IIA against those of other algorithms. In addition, Fig. 10 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 the average makespan very rapidly; the best global makespan (52) was achieved after 25 generations.

In addition, IIA considered various parameter combinations. Determining the appropriate parameters has important effects on the quality of the solutions and reduces the probability of avoiding premature convergence. As such, identifying the appropriate parameter combinations by analyzing different parameter combinations, specifically the clonal selection rates (C_r), local explorative mutation rates (R_{eesm}), and population sizes (pop_N), were investigated. The details of different parameter combination results are graphically shown in Figs. 11(a), (b), and (c). The pop_N value used are 50, 75, 150, 300; R_{eesm} value used are 0.05, 0.1, 0.15, and 0.3; and C_r used 0.25, 0.45, 0.65, and 0.75. Figure 11(a) shows the relative average deviation of makespan when different combinations of pop_N and C_r values were used. On the other hand, Fig. 11(b) shows the relative average deviation of makespan when different combinations of pop_N and R_{eesm} values were used. Figure 11(c) shows the relative average deviation of makespan when different combination of R_{eesm} and C_r values were used.

From the overall view of Fig. 11(a), fluctuation behaviour is observed when pop_N value is 50, 150, and 300. However, steady behavior is observed when pop_N value is 75, in which higher C_r value resulted in higher deviation values. However, suggested C_r values are 0.45, 0.65, and 0.75, which were combined with pop_N values of 50, 150, and 50, respectively. On the other hand, Fig. 11(b) revealed stochastic behavior with different values of pop_N. From the overall view of Fig. 11(b), the best values for R_{eesm} are 0.05 and 0.15 which suitable with pop_N values of 300 and 75, respectively. Lastly, Fig. 11(c) shows a steady decreasing trend of relative average deviation with increasing C_r values. Although fluctuating values were observed with different R_{essm} values, Fig. 11(c) suggests that R_{eesm} and C_r value of 0.1 and 0.75, respectively, produce the lowest relative average makespan deviation.

Arguably, different combinations of parameters of C_r, R_{eesm}, and pop_N values produced different outputs. However, the short-listed parameters provide insight on how IIA may have behaved differently. First, a high value of C_r (e.g., 0.65 and 0.75) may be coupled with low value pop_N (e.g., 50) and R_{eesm} (e.g., 0.1) to consistently produce the best makespan value. This scenario is possible as higher C_r value involves selecting higher percentage of population to be cloned and hyper-mutated (global and uniform local mutation) in small pop_N, where low number of R_{eesm} indicates that a small number of that population had explored different search spaces and relatively "escaped" possible local optima. On the other hand, a high value of R_{eesm} (e.g., 0.15) may shows some anomalies at a lower value of pop_N (e.g., 75) where instead of producing the worst output, the best makespan was obtained. This scenario is possible because diversity of solution is actively maintained, which drives the IIA to search other regions of the search space that have not been explored before. However, stochastic output may not be a feasible choice in manufacturing settings where the quality and productivity is the main concern. Thus, to summarize the best parameters out of the discussed ones, the best values were $150 \leq pop_N \leq 300$, $0.65 \leq C_r \leq 0.75$, and $0.05 \leq R_{eesm} \leq 0.1$.

The scheduling problem in FMSDS subject to machine maintenance is considered as another alternative to reduce cost and to increase overall productivity. This 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 amalgamating PM policy can maintain overall system performance while optimize production scheduling plan and reduce possible machine unavailability. The obtained results indicate that IIA produced a relatively more satisfactory solution than other meta-heuristic algorithms applied in a similar field. As in the work of Jia *et al.* [8], we observed diversity and quick solution evaluation because of the simplified encoding scheme of IIA. The greedy decoding scheme always guarantees a superior solution, thereby improving solution quality in each evaluation process. Therefore, IIA was found suitable and competitive in solving scheduling problem in FMSDS subject to machine maintenance.

The applicability and potential of IIA in solving the problem identified in this paper may be achieved by further improving the following aspects:

1) Given the stochastic nature of IIA parameters, an extension with an artificial neural-network can be developed to find system-specific parameters or operating strategies

TABLE V
COMPARISON BETWEEN THE RESULTS OF THE FIRST AND SECOND EXPERIMENTS.

Experiment 1									
Data Name	ACO	*dev*(%)	GADG 1,2,3	*dev*(%)	MGADG	*dev*(%)	IGA	*dev*(%)	IIA
fjs01	42	+33.33	36	+22.22	35	+20.00	35	+20.00	**28**
fjs02	n.a.	n.a.	227	+11.01	n.a.	n.a.	n.a.	n.a.	**202**
dfjs01a	n.a.	n.a.	42	+16.67	n.a.	n.a.	37	+5.41	**35**
dfjs01b	n.a.	n.a.	122	+30.33	93	+8.60	n.a.	n.a.	**85**

Experiment 2					
Data Name	MGA	*dev*(%)	IGA	*dev*(%)	IIA
Mt06	55	+9.09	55	+9.09	**50**
Mt10	972	+8.64	930	+4.52	**888**
Mt20	1207	+24.28	1172	+22.01	**914**
Average improvement		+14.00		+11.87	

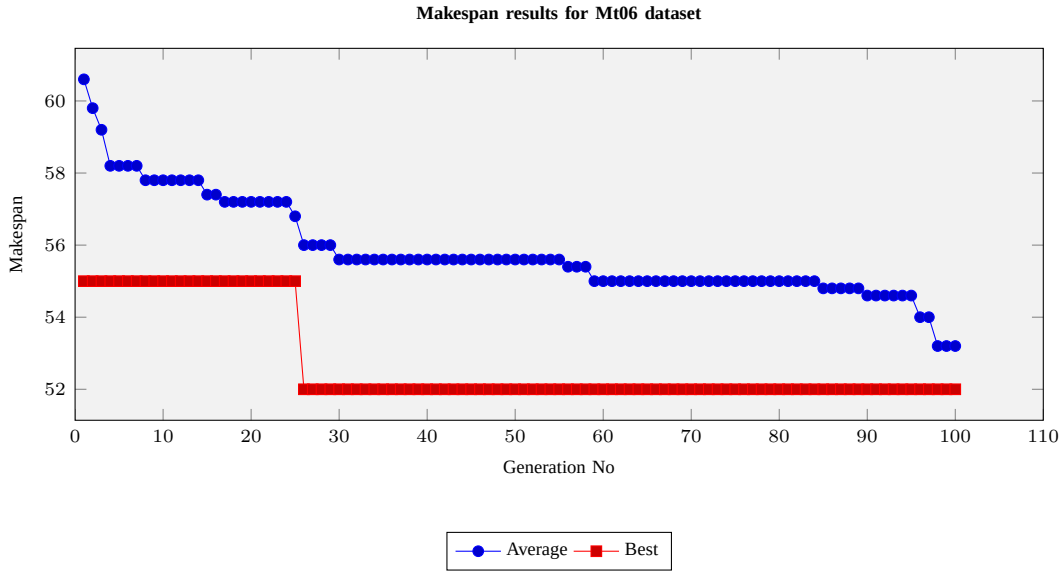

Fig. 10. Makespan results for Mt06 dataset.

or develops an expert system for obtaining scheduling knowledge in FMSDS environment.

2) Rescheduling strategies can be incorporated in IIA to improve solution quality and system state in a real-time operation, which can enhance productivity as a result..
3) Incorporating IIA with an efficient machine maintenance strategy can improve solution reliability and quality.
4) Worst-case scenarios (i.e., machine breakdown) can be simulated to further test IIA capabilities in a real-time environment.
5) A systematic methodology can be developed to add additional values to the major IIA operators (e.g., hyper-mutation, affinity maturation/clonal selection, and adaptive memory) specific to the scheduling problem.
6) Inclusion of other hardware elements of the manufacturing system can be considered to develop an integrated scheduling task.

VI. CONCLUSION

This paper proposed the IIA approach to solve scheduling problems in FMSDS subject to machine maintenance. The relative deviation of the results of the proposed IIA is better than that of other algorithms used in a similar situation. With a relative deviation in the range of 4% to 33%, the proposed IIA performs better and has superior optimization capabilities. These promising results pave the way for further extension of this work into a more complex and challenging environment. However, the datasets and benchmarks obtained from literature, merely serve as models of a real-world manufacturing problem, which is substantially complex and challenging in nature. As such, achieving conceivable results that address an actual manufacturing problem remains far from reality and actual implementation. Future research will present an extended design of the IIA approach, consider multi-objective performance, compare different implemented

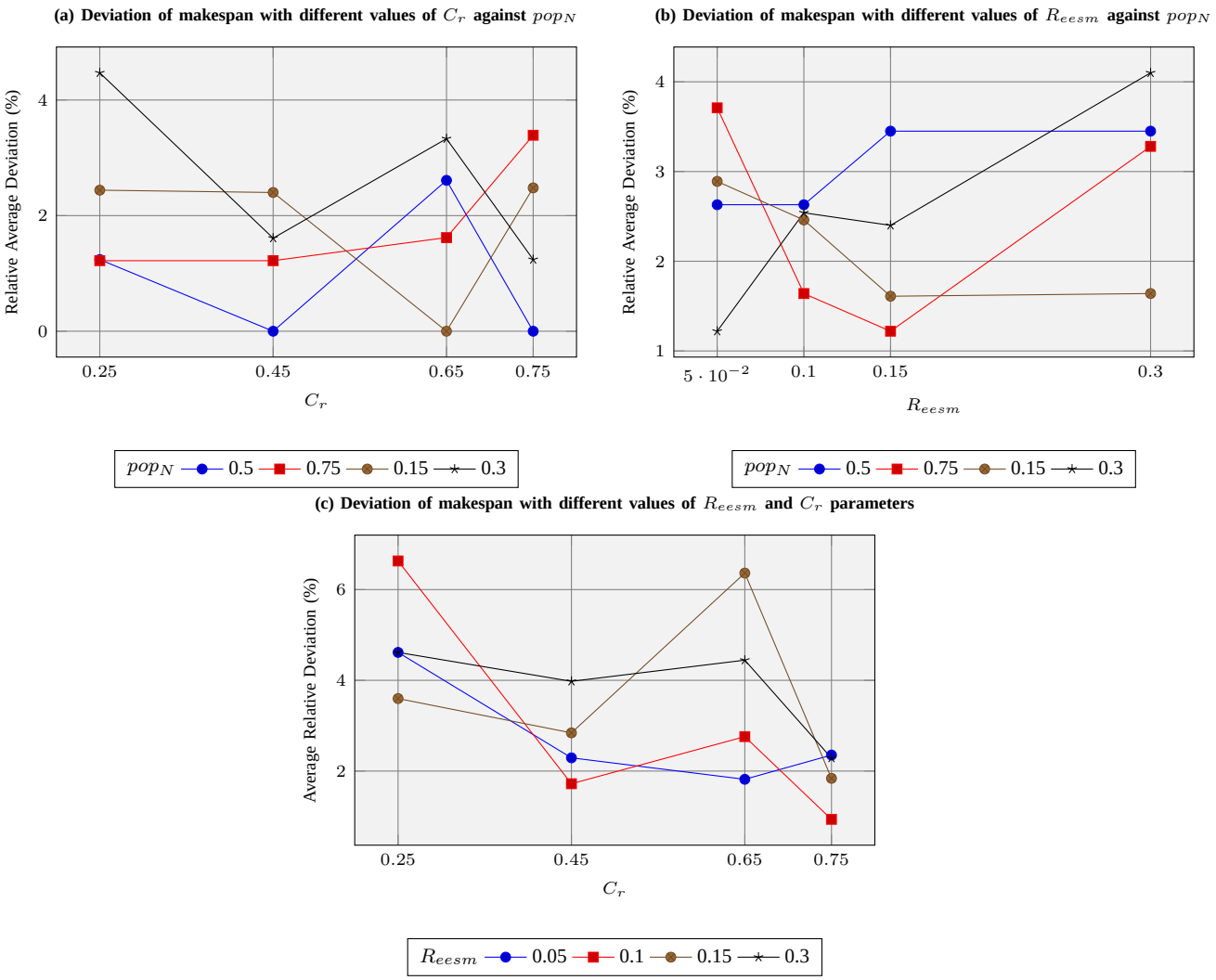

Fig. 11. Parameter analysis of IIA.

maintenance strategies, include larger jobs and factory data, and provide comprehensive parameters and analytical results.

ACKNOWLEDGMENT

The main author wishes to thank Universiti Sains Malaysia for its support in the completion of the present research from its inception.

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," *Intelligent Systems, IEEE*, vol. 17, no. 1, pp. 88–94, 2002.

[6] H. Jia, J. Fuh, A. Nee, and Y. Zhang, "Integration of genetic algorithm and gantt chart for job shop scheduling in distributed manufacturing systems," *Computers & Industrial Engineering*, vol. 53, no. 2, pp. 313–320, 2007.

[7] 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, p. 35573563, 2011.

[8] 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, pp. 351–362, 2003.

[9] 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.

[10] 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.

[11] D. Lee and F. DiCesare, "Scheduling flexible manufacturing systems using petri nets and heuristic search," *Robotics and Automation, IEEE Transactions on*, vol. 10, no. 2, pp. 123–132, 1994.

[12] J. Paulli, "A hierarchical approach for the fms scheduling problem," *European Journal of Operational Research*, vol. 86, no. 1, pp. 32–42, 1995.

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

[15] 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.

[16] S. Wadhwa, A. Prakash, and S. Deshmukh, "A knowledge based ga approach for fms scheduling," in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 2. Citeseer, 2009.

[17] S. Burnwal and S. Deb, "Scheduling optimization of flexible manufacturing system using cuckoo search-based approach," *The International Journal of Advanced Manufacturing Technology*, pp. 1–9, 2012.

[18] F. Chan, S. Chung, and L. Chan, "A study of distributed scheduling problem with machine maintenance," in *Cybernetics and Intelligent Systems, 2006 IEEE Conference on*. IEEE, 2006, pp. 1–6.

[19] M. Gopalakrishnan, S. Mohan, and Z. He, "A tabu search heuristic for preventive maintenance scheduling," *Computers & industrial engineering*, vol. 40, no. 1-2, pp. 149–160, 2001.

[20] C. Pereira, C. Lapa, A. Mol, and A. Da Luz, "A particle swarm optimization (pso) approach for non-periodic preventive maintenance scheduling programming," *Progress in Nuclear Energy*, vol. 52, no. 8, pp. 710–714, 2010.

[21] H. Xue, S. Wei, and L. Yang, "Preventive maintenance scheduling of fms equipment based on improved ant colony algorithm," in *Industrial Engineering and Engineering Management (IE&EM), 2011 IEEE 18Th International Conference on*. IEEE, 2011, pp. 1128–1131.

[22] S. 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.

[23] A. Prakash, N. Khilwani, M. Tiwari, and Y. Cohen, "Modified immune algorithm for job selection and operation allocation problem in flexible manufacturing systems," *Advances in engineering software*, vol. 39, no. 3, pp. 219–232, 2008.

[24] K. Woldemariam and G. Yen, "Vaccine-enhanced artificial immune system for multimodal function optimization," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 40, no. 1, pp. 218–228, 2010.

[25] G. Wojtyla, K. Rzadca, and F. Seredynski, "Artificial immune systems applied to multiprocessor scheduling," *Parallel Processing and Applied Mathematics*, pp. 904–911, 2006.

[26] L. De Castro and F. Von Zuben, "Learning and optimization using the clonal selection principle," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 3, pp. 239–251, 2002.

[27] V. Cutello, G. Nicosia, M. Romeo, and P. Oliveto, "On the convergence of immune algorithms," in *Foundations of Computational Intelligence, 2007. FOCI 2007. IEEE Symposium on*. IEEE, 2007, pp. 409–415.

[28] J. Al-Enezi, M. Abbod, and S. Alsharhan, "Artificial immune systems-models, algorithms and applications," *International Journal*, 2010.

[29] D. Dasgupta and S. Forrest, "Artificial immune systems in industrial applications," in *Intelligent Processing and Manufacturing of Materials, 1999. IPMM'99. Proceedings of the Second International Conference on*, vol. 1. IEEE, 1999, pp. 257–267.

[30] H. Yu, "Optimizing task schedules using an artificial immune system approach," in *Proceedings of the 10th annual conference on Genetic and evolutionary computation*. ACM, 2008, pp. 151–158.

[31] S. Darmoul, H. Pierreval, and S. Gabouj, "Scheduling using artificial immune system metaphors: a review," in *Service Systems and Service Management, 2006 International Conference on*, vol. 2. IEEE, 2006, pp. 1150–1155.

[32] O. Kilic and Q. Nguyen, "Application of artificial immune system algorithm to electromagnetics problems," *Progress In Electromagnetics Research*, vol. 20, pp. 1–17, 2010.

[33] C. Chu, M. Lin, G. Liu, and Y. Sung, "Application of immune algorithms on solving minimum-cost problem of water distribution network," *Mathematical and Computer Modelling*, vol. 48, no. 11, pp. 1888–1900, 2008.

[34] L. De Castro and F. Von Zuben, "The clonal selection algorithm with engineering applications," in *Proceedings of GECCO*, vol. 2000, 2000, pp. 36–39.

[35] C. Coello and N. Cort'es, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, 2005.

[36] N. Khilwani, A. Prakash, R. Shankar, and M. Tiwari, "Fast clonal algorithm," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 1, pp. 106–128, 2008.

[37] S. Hofmeyr, *An interpretative introduction to the immune system*. New York: Oxford University Press, 2000, vol. 3.

[38] 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.

[39] 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.

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

[41] 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.

[42] U. Ojha and M.-Y. Chow, "An analysis of artificial immune system and genetic algorithm in urban path planning," in *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, 2010, pp. 1064–1069.