International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:6, No:9, 2012

Solving Part Type Selection and Loading Problem in Flexible Manufacturing System Using Real Coded Genetic Algorithms – Part I: Modeling

Wayan F. Mahmudy, Romeo M. Marian, Lee H. S. Luong

***Abstract*—**This paper and its companion (Part 2) deal with modeling and optimization of two NP-hard problems in production planning of flexible manufacturing system (FMS), part type selection problem and loading problem. The part type selection problem and the loading problem are strongly related and heavily influence the system's efficiency and productivity. The complexity of the problems is harder when flexibilities of operations such as the possibility of operation processed on alternative machines with alternative tools are considered. These problems have been modeled and solved simultaneously by using real coded genetic algorithms (RCGA) which uses an array of real numbers as chromosome representation. These real numbers can be converted into part type sequence and machines that are used to process the part types. This first part of the papers focuses on the modeling of the problems and discussing how the novel chromosome representation can be applied to solve the problems. The second part will discuss the effectiveness of the RCGA to solve various test bed problems.

***Keywords*—**Flexible manufacturing system, production planning, part type selection problem, loading problem, real-coded genetic algorithm

I. INTRODUCTION

RAPID market changes (changing customer needs), peaks in demand for product quantity (e.g. new gadgets – tablet PC, mobile phones, etc.), concerns for product quality, and requirements to dramatically increase product mix have forced manufacturing industries to enhance their flexibility. Flexible manufacturing system (FMS) is designed to cope with these conditions by using high technologies and automation in transfer lines which enable factories to reconfigure rapidly to produce a variety of products by using same resources [1-3]. Due to the high investment required, higher resources utilization (close to 100%) must be achieved and this issue can be resolved by establishing a good production planning. This planning will also increase productivity by maximizing system throughput (number of produced part types in each batch) and enable early return on investment.

The areas of research related with the FMS can be grouped into design and operational problem, usually addressed by using hierarchical approaches [4].

The design problem focuses on decision of system specification when the need for automation and flexibility is started to achieve desired goals. Several decisions must be established in this stage such as system hardware and software, hierarchy of control mechanism, FMS configuration or FMS type, and determine the range of part types to be produced. Operational problem is related with planning, scheduling and control of FMS [5]-[6]. All the research problems in FMS can be described in Figure 1. This paper focuses on the planning problem.

The planning problem in FMS is related with the arrangement of parts and technological equipments such as tools, fixtures and pallets, and the determination of the type and quantity of the products which are made before starting production [7]-[8].

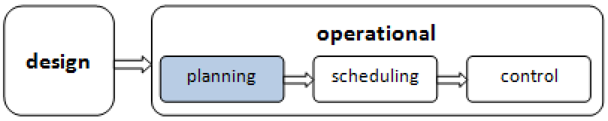

Fig. 1 Research area in FMS

The planning problems can be divided into two sub problems, an aggregate production planning and a short term planning (or a production planning) [6]. The aggregate production planning produces a master schedule containing part mix, production rates and lot sizes. The production planning gives interface between aggregate production planning and daily operation of the FMS. There are several issues in the production planning stage such as part type selection problem, machine grouping problem, production ratio problem, resource allocation problem, and loading problem [5]-[9].

The part type selection problem and the loading problem are parts of the production planning problems which are strongly related and heavily influence the system's efficiency [10][11]. The part type selection problem deals with selection of set of part types (products) which must be produced immediately from a number of part types as there are different due dates, limited number of machines, limited tool magazines capacity of each machine and limited number of tools.

The loading problem is concerned with allocation of operations for selected part types and loading appropriate tools to the machines [5]-[12]. Solving part type selection and machine loading problem simultaneously will produce a better solution as higher throughput of the FMS is achieved while keeping the balance of machines' workload [3].

W. F. Mahmudy is a PhD student at School of Advanced Manufacturing & Mechanical Engineering, University of South Australia and a lecturer at Department of Computer Science, Brawijaya University (UB), Indonesia (e-mail: wayan_firdaus.mahmudy@mymail.unisa.edu.au, wayanfm@ub.ac.id).

R. M. Marian, PhD is Senior Lecturer and Program Director at School of Advanced Manufacturing & Mechanical Engineering, University of South Australia, Australia (e-mail: romeo.marian@unisa.edu.au).

L. H. S. Luong is a professor at School of Advanced Manufacturing & Mechanical Engineering, University of South Australia, Australia (email: lee.luong@unisa.edu.au).

Here, the efficient allocation of production resources will be achieved.

Solving the part type selection problem and the machine loading problem simultaneously require a good approach to achieve a good result on reasonable amount of time. Genetic algorithms (GAs) are regarded as the powerful method to solve a complex problem with a large search space [13]. GAs have an ability to escape from local optima as they can jump randomly from one sequence to another sequence [14]. The power of GAs to solve various complex problems has attracted a lot of researchers to do research in this area. However, a simple GAs is insufficient to solve any complex problems in engineering. A proper representation, developing appropriate search operators and hybridizing it with other methods are an important key for its successful implementation and becoming challenging tasks [15][16]. In addition, a good strategy to avoid premature convergence which produces local optimum solution should be developed [17]. This paper attempts to develop a new representation using an array of real numbers which could produce good solutions efficiently by using simple genetic operators. The novel proposed chromosome representation is designed to produce only feasible solutions which minimize a computational time needed by GAs to push its population toward feasible search space or repair infeasible chromosomes.

II. LITERATURE REVIEW

The part type selection problem and the loading problem in the FMS environments have received significant attentions from researchers who proposed a various approaches. Tabucanon, Batanov & Basu [18] used simulation to evaluate the solution of part type selection and loading problem produced by batching approach. They formulated mathematical programming method to maximize the number of part types in each batch. Denizell & Sayin [19] developed a mathematical programming model to solve the part type selection problem that considering due dates. Choudhary, Tiwari & Harding [20] addressed the problems by using a GAs with chromosome differentiation. A sexual differentiation of the chromosomes was applied to maintain a diversity of the population and explore the search space extensively. Tiwari et al. [12] proposed a constraints-based fast simulated annealing algorithm to solve a combination of part-type selection and operation allocation on machines. Their proposed algorithm which was performed by a combination of a GA and a simulated annealing (SA) had a capability to escape from local optimum and provide good solutions. Biswas & Mahapatra [10] proposed modified particle swarm optimization (PSO) to solve machine loading problems. This algorithm attempted to maintain the balance of the system while regarding the occurrence of technological constraints such as the availability of machining time and tool slots.

Ponnambalam & Kiat [21] also used PSO to solve machine loading problems. This algorithm is equipped with two local search method to improve the solution quality. They applied two objectives, minimizing system unbalance and maximizing system throughput, while satisfying the technological constraints. Tiwari, Kumar Jha & Bardhan Anand [11] developed a combinatorial auction-based heuristic for multi-agent system to solve the problems. This approach was used to deal with a huge search space of the part type selection and machine loading problems. Even though all these researchers reported promising results, several simplicities were made to reduce the complexity of the problems.

Part type selection and machine loading are NP-hard problems [10]. The complexity of the problems is harder when flexibilities of operations are considered. For example, each part has alternative routes (routing flexibility) which refer to a possibility of operation is processed on alternative machines with alternative tools. For simplicity, Tabucanon, Batanov & Basu [18] did not consider the routing flexibility. Denizell & Sayin [19] and Pacciarelli [22] considered FMS that consists of all general-purpose machines where the functionality of a machine is only determined by the set of tools loaded in their tool magazine. Here, each part is only processed by one machine. Furthermore, Swarnkar & Tiwari [23], Choudhary, Tiwari & Harding [20], Biswas & Mahapatra [10], Ponnambalam & Kiat [21], Prakash et al. [24] and Tiwari, Kumar Jha & Bardhan Anand [11] did not mention specific tool types and its availability; they only mentioned the number of slots needed by the tools. In contrast, this paper considers machine and tool flexibility and also limited numbers of tool types.

III. PROBLEM DESCRIPTION

This paper considers a FMS which consists several computer numerically controlled (CNC) machines equipped with a tool magazine which has limited tool slot capacity. The machines can perform different operations when they are equipped with different tools. A limited number of tools are available and each tool requires a number of slots when it is assigned to a machine. When several jobs (part types) arrive, the system must select a set of part types which must be produced immediately as there is a limitation of machines and its tool slot capacity and tools availability. This approach is considered as batching approach as all part type should be grouped into several production batches [18].

Each part type has a production requirement in form of sequence of operations. Each operation can be processed on several alternative machines with several alternative tools. This paper also considers unrelated machines approach where time needed for parts' operations depend on the assigned machine.

A. Subscripts

$p = 1,\dots,P$	part type
$o = 1,\dots,O_p$	operation of part type p
$t = 1,\dots,T$	tool type
$m = 1,\dots,M$	machine type

B. Parameters

MS_m = tool slot capacity of machine m

TN_t = number of tools type t
TS_t = number of slots required by tool type t
PS_p = batch size of part type p
PV_p = value (price) of part type p
MOP_{po} = set of possible machines on which operation o of part type p can be performed
$TM_{pomt} = \{1,0\}$: 1 if tool t is required for processing operation o of part type p on machine m
T_{pom} = processing time of operation o of part type p on machine m

C. Decision variables

$X_p = \{1,0\}$: 1 if part type p is selected in the batch
$X_{pom} = \{1,0\}$: 1 if machine m is selected to process operation o of part type p
$Y_{mt} = \{1,0\}$: 1 if tool t is loaded to the machine m

D. Objectives

A various objectives had been considered in the references such as maximizing system throughput [3][12][14][18-21][23-26], maintaining the balance of the system [3][10][12][14][20][21][23-27], minimizing part movement [27], minimizing tool changeovers [27] and minimizing production cost [28]. However, most of references considered two common objectives, maximizing system throughput and maintaining the balance of the system (minimizing system unbalance). System throughput and system unbalance can be calculated in different ways as follows:

1. Maximizing system throughput

$$\text{Maximize } : \sum_{p=1}^{P} X_p \tag{1}$$

$$\text{Maximize } : \sum_{p=1}^{P} X_p PS_p PV_p \tag{2}$$

Maximizing system throughput can be achieved by maximizing the number of selected part types in each batch which can be expressed as in (1). This objective means minimizing a time lost for tools changeover. Tabucanon, Batanov & Basu [18], Kumar & Shanker [3] used this objective function. Another way to maximize system throughput is by maximizing the value (price or profit) or the sum of batch size (if all part types have equal price or profit) of selected part types in each batch as shown in (2). Kumar & Shanker [3], Choudhary, Tiwari & Harding [20], Prakash et al. [24], Ponnambalam & Kiat [21], and Yogeswaran, Ponnambalam & Tiwari [25] used this objective function.

2. Maintaining the balance of the system

$$\text{Minimize } : \sum_{m=1}^{M} \left| W_m - \overline{W} \right|$$
$$\text{where } \; W_m = \sum_{p=1}^{P} \sum_{o=1}^{O_p} X_{pom} T_{pom} \tag{3}$$
$$\text{and } \; \overline{W} = \left(\sum_{m=1}^{M} W_m \right) \Big/ M$$

$$\text{Maximize } : \max_{m=1,\dots,M} \left\{ \sum_{p=1}^{P} \sum_{o=1}^{O_p} X_{pom} T_{pom} \right\} \tag{4}$$

$$\text{Minimize } : \sum_{m=1}^{M} \left| SP_m - W_m \right| \tag{5}$$

$$\text{Minimize } : \sum_{m=1}^{M} \left(SP_m - W_m \right) \tag{6}$$

Maintaining the balance of the system can be achieved by minimizing system unbalance as expressed in (3) where W_m is workload of machine m and $\overline{W}$ is the average machine workload. Seok Shin, Park & Keun Kim [27] used this objective function. Another way to minimize system unbalance is by minimizing a maximum machine's load as in (4). If length of scheduling period for each machine (SP_m) is determined in advance and overloading of the machines is allowed, the system unbalance may be expressed as in (5). Mukhopadhyay, Midha & Krishna [26], Biswas & Mahapatra [10] used this objective function. However, if overloading of the machines is not allowed, Equation (6) is used. Here, length of scheduling period becomes a maximum machine workload. Choudhary, Tiwari & Harding [20], Biswas & & Mahapatra [10] used this objective function.

E. Constraints

While minimizing system unbalance and maximizing system throughput, several technological constraints must be satisfied as follows:

$$\sum_{o=0}^{O_p} \sum_{m=1}^{M} X_{pom} = O_p X_p \qquad p = 1,\dots,P \tag{7}$$

$$\sum\nolimits_{m \in MOP_{po}} X_{pom} = X_p \qquad p = 1,\dots,P \qquad o = 1,\dots,O_p \tag{8}$$

$$\begin{gathered} Y_{mt} = X_{pom} TM_{pomt} \\ p = 1,\dots,P \qquad o = 1,\dots,O_p \qquad m = 1,\dots,M \qquad t = 1,\dots,T \end{gathered} \tag{9}$$

$$\sum_{m=1}^{M} Y_{mt} \le TN_t = O_p X_p \qquad t = 1,\dots,T \tag{10}$$

$$\sum_{t=1}^{T} Y_{mt} TS_t \le MS_m = O_p X_p \qquad m = 1,\dots,M \tag{11}$$

$$\sum_{p=1}^{P} \sum_{o=1}^{O_p} X_{pom} T_{pom} < SP_m \qquad m = 1,\dots,M \tag{12}$$

Constraint (7) guarantees that if a part type is selected, all its operations must be performed. This constraint states that operation assignments are equal to the total operations required.

Constraint (8) states that each operation of selected part types must be completed on one machine. As there is possibility of operation can be processed on alternative machines, the machine must be determined and the operation must be processed by the chosen machine. Constraint (9) guarantees that if a machine is selected to process an operation of a part type, all the tools needed must be loaded to the machine. Constraint (10) ensures that the number of tools loaded to the machines must not exceed its availability.

Constraint (11) ensures that the number of tool slots used on a machine must not exceed the machine's tool slot capacity. Constraint (12) is only used if length of scheduling period for each machine (SP_m) is determined in advance and overloading of the machines is not permitted.

IV. MODELING USING GA

GAs are general purpose search algorithms which imitate a natural evolution process. Candidate solutions are represented by chromosomes which evolve over time (generations) through reproduction and stochastic selection. Along generations these chromosomes become better (with higher fitness value) and at the final generation the best chromosome can be decoded as a near optimum solution [29]. This section describes how real-coded genetic algorithm is used to solve part type selection and loading problem.

A. Chromosome representation

A suitable chromosome representation determines the successful implementation of genetic algorithms [30]. This paper uses real number representation so GAs which uses this representation can be called real-coded GAs (RCGA). A chromosome is a vector of real number whose size is same with the number of part types. This representation usually was used to solve optimization problems on continuous domains [31]. However, a simple implementation of its operators (crossover and mutation) and possibility to decode one real number into several meanings (part type's index and its several operations) become the main reason to use this representation for solving part type selection and loading problem.

The construction of a chromosome in our RCGA is shown in Table I. Each element of the chromosome $X=(x_1,x_2,\ldots,x_p)$ corresponds to the continuous position values for p number of part-types. The value of x_i is maintained between 0 and $2^{\text{opMax} \times \text{bitMac} + \text{bitPart}}$. *opMax* is maximum number of operations of each part type. *bitMac* is number of bits required to represent a binary number which has largest value of maximum number of alternative machines of each operation. For example, the maximum number of alternative machines of each operation is 5. Therefore, the minimum bits required to represent a binary number between 0 and 5 is 3. *bitPart* is number of bits required to represent a binary number which has largest value of number of part types. x_i is stored and treated as a real number when genetic operators (crossover and mutation) are applied.

TABLE I
CHROMOSOME CONSTRUCTION

part type	1	2	3	4	5	6	7	8
x	2772	7779	5129	7981	6215	977	9969	1654
part type sequence	6	8	1	3	5	2	4	7
sorted x	977	1654	2772	5129	6215	7779	7981	9969

However, x_i will be converted (rounded) to a nearest integer value when decoding operation is performed.

A smallest position value (SPV) rule is used to get part types sequence. By sorting x in ascending order, we obtain the sequence of part types that are selected for the current batch. To determine which machines are used to process part types' operations, each element of X is decoded into a binary number. Suppose part type 8 has 3 operations, operation 1 can be processed on machines 2 or 3 or 5, operation 2 can be processed on machine 1 or 2 or 6 or 7, and operation 3 can be processed on machine 3 or 4. To choose machines used for processing of operations of part type 8, x_2=1654 is converted into a binary number $(11001110110)_2$. Suppose *bitMac* is 3. For the first operation, we use 3 bits at the right side $(110)_2$ which is equal to 6. As there are 3 possible machines (n=3), we apply the following formula:

$$\textit{machine index} = 6 \bmod n + 1 = 6 \bmod 3 + 1 = 1$$

mod is modulus operator which gives the remainder of a division. Therefore, the first operation of part type 8 is performed on the first possible machine that is machine 2. By using the next 3 bits and applying the same rule, we obtain that part type 8 is sequentially processed on machines 2, 6 and 4.

After determining the machines for operations, required tools are assigned to the machines. At this step, all the constraints such as availability of tools and empty slots on the machines are checked. For example, after choosing part types 6, 8, 1 and 3 according to the part type sequence as shown in Table 1, adding part type 5 to the solution violate the constraints. Therefore, the chromosome states that only part types 6, 8, 1 and 3 are selected for the current batch and the objective functions of the problem are calculated based on these selected part types.

Note that the proposed representation produces only feasible solutions and guarantees that only required tools assigned to the machines. This effort will minimizes a computational time which usually needed by GAs to push its population toward a feasible search space or repeatedly repair infeasible chromosomes [32].

B. Fitness function

The objective functions of the optimization problem must be converted to a fitness function which is used to measure the goodness of the solution. For example, Equation (2) is used to measure system throughput, Equation (2) should be converted into (13) to produce value between 0 and 1.

$$f_1 = \left(\sum_{p=1}^{P} X_p PS_p PV_p \right) \Bigg/ \left(\sum_{p=1}^{P} PS_p PV_p \right) \tag{13}$$

Furthermore, Equation (5) is used to measure system unbalance. Here, length of scheduling period for each machine (SP_m) is determined in advance and overloading of the machines is permitted. Minimizing (5) can be converted as maximizing (14) as follow:

$$f_2 = 1 - \left(\sum_{m=1}^{M} \left| SP_m - W_m \right| \right) \Bigg/ \sum_{m=1}^{M} SP_m \tag{14}$$

Finally, the fitness function can be formulated as follow:

$$\text{Maximize } : F = \alpha_1 f_1 + \alpha_2 f \qquad (15)$$
$$\alpha_1, \alpha_1 : \quad \text{weigthed parameters}$$

C. Initialization of population

pop_size of chromosomes are created as an initial population. Here, x_i is generated randomly within its range.

D. Reproduction

On every generation, new chromosomes (*offspring*) are produced by using crossover operator and mutation operator. The number of new chromosomes produced is determined by *crossover-rate* (*cr*) and *mutation-rate* (*mr*) parameters. For example, if population size is *pop_size* then there are *pop_size*×*cr* offspring produced by crossover operator and *pop_size*×*mr* offspring produced by mutation operator for each generation. Parents for these reproduction operations are randomly and uniformly chosen from the population.

This paper uses two crossover methods, *flat-crossover* [33] and *extended-intermediate-crossover* [34]. Let $P_1=(p_1^1,\dots,p_n^1)$ and $P_2=(p_1^2,\dots,p_n^2)$ are two selected chromosomes as parents for crossover. *Flat-crossover* produces offspring $O=(o_1,\dots,o_n)$ by generating a random number o_i on interval $[p_i^1,\dots,p_i^2]$. *Extended-intermediate-crossover* uses a formula $o_i=p_i^1+\alpha_i(p_i^2-p_i^1)$, where α_i is randomly generated on interval [–0.25, 1.25]. These crossover methods are randomly chosen in each generation.

Random exchange mutation which is usually applied in permutation representation is used. This mutation works by selecting two genes randomly and exchanging their positions. We also develop mutation method for real number representation, *simple-random-mutation*. If $P=(p_1,\dots,p_n)$ is selected parent for mutation then offspring $O=(o_1,\dots,o_n)$ is produced by applying a formula $o_i=p_i+\alpha_i$, where α_i is randomly generated on interval [–0.1, 0.1]. These mutation methods are randomly chosen in each generation.

All offspring produced in this stage are placed in offspring pool.

TABLE II
TOOL TYPES' AVAILABILITY

tool type	1	2	3	4	5	6	7	8	9	10
availability	2	2	2	2	2	3	3	3	3	3
number of slot needed	3	3	4	4	5	5	4	4	3	3

E. Selection

Selection procedure is used to select *pop_size* chromosomes from current population (parents) and offspring pool to perform the next generation. Four common selection methods will be examined to determine which method is most suitable for the RCGA. These selection methods are:

1. Roulette wheel selection

Each chromosome from current population (parents) and offspring pool has probability to be selected according to its fitness value. Here, a cumulative probability is calculated and a random number is generated to select the chromosome.

2. Binary tournament selection

One chromosome from current population and one chromosome from offspring pool are randomly chosen and compared. The best one will be selected.

3. Elitist selection

All chromosomes from current population (parents) and offspring pool are placed in one pool and sorted according their fitness value. *pop_size* best chromosomes are selected.

4. Replacement selection

Each chromosome in offspring pool will be selected to replace its parent if it has a better fitness value than its parent. If the child is produced by crossover operator (by using two parents) then the child will replaces the worst parent.

Note that the binary tournament, elitist and replacement selection guarantee that the best chromosome is always passed to the next generation.

F. Overall RCGA cycle

The overall RCGA cycle is shown as follow:

Step 0: Setting GA parameters
Parameters: population size *pop_size*, crossover rate *cr*, mutation rate *mr*, maximum number of generations *max_gen*, weighted parameters (α_1 and α_2) for fitness function.
Step 1: Initialization
Let generation *gen*=0.
Generate *pop_size* of random chromosomes.
Step 2: Reproduction
Produce *pop_size*×*cr* offspring by using crossover operator and *pop_size*×*mr* offspring by using mutation operator.
Step 3: Selection
Select *pop_size* chromosomes from population and offspring pool for the next generation.
Step 4: Let *gen* ← *gen* + 1.
If gen=*max_gen* go to Step 2, else Stop.

TABLE III
PART TYPE PRODUCTION REQUIREMENT

part type	batch size	value ($)	op	mac	time (seconds)	tools		
1	30	5	1	1	30	1	2	3
			2	1	20	4	5	
				3	20	2	3	
			3	3	30	3	4	
2	30	3	1	1	40	1	2	
			2	2	20	3	4	
			3	3	30	5	6	7
3	30	2	1	1	30	6	7	8
				3	40	8	9	10
			2	2	40	1	10	
				3	30	1	10	
			3	1	20	1	2	
4	30	1	1	3	30	9	10	
				2	20	9	10	

			2	2	30	6	7	
				1	40	6	7	
			3	1	30	3	4	
5	40	4	1	1	40	1	2	3
			2	2	40	4	5	
				3	40	4	5	
6	40	3	1	3	20	7	8	
			2	2	50	9	10	
			3	1	10	3		
7	40	2	1	2	20	3	4	
			2	2	30	1	2	
				3	40	8	9	
8	40	5	1	1	50	1	2	3
				3	40	8	9	10
			2	2	30	4	5	

op:operation, mac:machine, ntool:number of tools required

V.RESULT

A simple problem set is given to demonstrate how the proposed RCGA solves the problem. There are 3 different machines which have tool slot capacity of 20, 15 and 20 respectively. Length of scheduling period for each machine (SP_m) is 4000 and overloading of the machines is allowed. Furthermore, as shown in Table 2, there are 10 tool types where each tool type requires several tool slots on machines' magazine.

Eight part types are ready to be produced as shown in Table 3. Each part type has specific production requirements. For example, part type 1 has 3 operations. Operation 2 can be processed on machine 1 or 3. Machine 1 needs 20 unit times for processing and tools 4, 5 are required. Note that the problem has machine and tool flexibility.

As shown in Table 3, maximum number of operations of each part type is 3 (*opMax*=3) and maximum number of alternative machines is 2 that requires 2 bits for a binary number (*bitMac*=2). Number of part types is eight that requires 4 bits for a binary number (*bitPart*=4). Therefore, the value of each element of chromosome is maintained between 0 and $2^{3\times2+4}$.

Several GAs parameters must be determined in advance as follows:

- Population size is 100.
- Crossover rate is 0.3.
- Mutation rate is 0.1.
- Maximum number of generations is 500.
- Weighted parameters (α_1 and α_2) are set equal to 1.

TABLE IV
THE BEST CHROMOSOME

x	793	603	439	1022	344	713	86	426

TABLE V
SEQUENCE OF SELECTED PART TYPES

part type	value	chosen machines
7	80	3 3
5	160	1 2
8	200	1 2
3	60	3 3 1
Throughput	500	

TABLE VI
MACHINES WORKLOAD

mac	workload	unbalance	number of slots	used slot	tools assigned
1	4200	200	20	10	1 2 3
2	3600	400	15	13	3 4 5
3	3700	300	20	13	1 8 9 10
System unbalance		900			

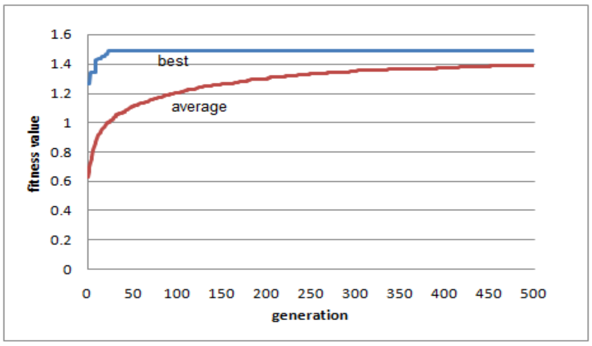

Fig. 2 The best and average fitness value

Crossover rate of 0.3 means that 100x0.3=30 offspring are produced by crossover operator for each generation. Similarly, mutation rate of 0.1 means that 100x0.1=10 offspring are produced by mutation operator for each generation. Replacement selection is used.

By using these parameters, the RCGA produces an optimum solution for the part type selection and loading problem in less than 1 second. The optimum solution is achieved after 23 generations. This optimum solution is checked by using branch-and-bound method. The best chromosome in Table 4 is converted to determine selected part types and its chosen machines for operations as shown in Table 5.

Machines' workload and tools assigned are shown in Table 6.

The increase of the best and average fitness value is depicted in Figure 2 which shows a fast convergence of the RCGA to optimality.

VI. CONCLUSION

Part type selection and loading problem with flexibilities of operations have been modeled in this paper. A simple problem set is given to demonstrate how the proposed RCGA solves the problem. A novel chromosome representation supported by suitable genetic operators enable the RCGA producing promising results in reasonable amount of time. In part 2, we will discuss about the effectiveness of each genetic operator and the quality of the results by using several test bed problems.

REFERENCES

[1] F.T.S. Chan and H.K. Chan, "A Comprehensive Survey and Future Trend of Simulation Study on Fms Scheduling," Journal of Intelligent Manufacturing, vol. 15, no. 1, pp. 87-102, 2004.

[2] I. Badr, "An Agent-Based Scheduling Framework for Flexible Manufacturing Systems," International Journal of Computer,

Information, and Systems Science, and Engineering, vol. 2, no. 2, pp. 123-129, 2008.

[3] N. Kumar and K. Shanker, "A Genetic Algorithm for Fms Part Type Selection and Machine Loading," International Journal of Production Research, vol. 38, no. 16, pp. 3861-3887, 2000.

[4] Y.Z. Mehrjerdi, "A Decision-Making Model for Flexible Manufacturing System," Assembly Automation, vol. 29, no. 1, pp. 32-40, 2009.

[5] K.E. Stecke, "Design, Planning, Scheduling, and Control Problems of Flexible Manufacturing Systems," Annals of Operations Research, vol. 3, no. 1, pp. 1-12, 1985.

[6] M. Ben-Daya, "Fms Short Term Planning Problems: A Review," in Manufacturing Research and Technology, A. Raouf and M. Ben-Daya, Eds., Elsevier. p. 113-139, 1995.

[7] Z. Binghai, X. Lifeng, and C. Yongshang, "A Heuristic Algorithm to Batching and Loading Problems in a Flexible Manufacturing System," The International Journal of Advanced Manufacturing Technology, vol. 23, no. 11, pp. 903-908, 2004.

[8] J. Venkateswaran, Y.-J. Son, and A. Jones. "Hierarchical Production Planning Using a Hybrid System Dynamic-Discrete Event Simulation Architecture," in Proceedings of the 2004 Winter Simulation Conference, 2004.

[9] S. Özpeynirci and M. Azizoglu, "Bounding Approaches for Operation Assignment and Capacity Allocation Problem in Flexible Manufacturing Systems," Computers & Operations Research, vol. 36, no. 9, pp. 2531-2540, 2009.

[10] S. Biswas and S. Mahapatra, "Modified Particle Swarm Optimization for Solving Machine-Loading Problems in Flexible Manufacturing Systems," The International Journal of Advanced Manufacturing Technology, vol. 39, no. 9, pp. 931-942, 2008.

[11] M.K. Tiwari, S. Kumar Jha, and R. Bardhan Anand, "Operation Allocation and Part Type Selection in E-Manufacturing: An Auction Based Heuristic Supported by Agent Technology," Robotics and Computer-Integrated Manufacturing, vol. 26, no. 4, pp. 312-324, 2010.

[12] M.K. Tiwari, S. Kumar, S. Kumar, Prakash, and R. Shankar, "Solving Part-Type Selection and Operation Allocation Problems in an Fms: An Approach Using Constraints-Based Fast Simulated Annealing Algorithm," IEEE Transaction on Systems, Man, and Cybernetics—Part A: Systems and Humans, vol. 36, no. 6, pp. 1170-1184, 2006.

[13] W. Shen, "Genetic Algorithms in Agent-Based Manufacturing Scheduling Systems," Integr. Comput.-Aided Eng., vol. 9, no. 3, pp. 207-217, 2002.

[14] M. Tiwari, J. Saha, and S. Mukhopadhyay, "Heuristic Solution Approaches for Combined-Job Sequencing and Machine Loading Problem in Flexible Manufacturing Systems," The International Journal of Advanced Manufacturing Technology, vol. 31, no. 7, pp. 716-730, 2007.

[15] M. Gen and R. Cheng, Genetic Algorithms and Engineering Optimization, New York: John Wiley & Sons, Inc., 2000.

[16] F. Rothlauf, Representations for Genetic and Evolutionary Algorithms, Berlin Heidelberg: Springer, 2006.

[17] M. Lozano and F. Herrera, "Fuzzy Adaptive Genetic Algorithms: Design, Taxonomy," Soft Computing, vol. 7, pp. 545–562, 2003.

[18] M.T. Tabucanon, D.N. Batanov, and S. Basu, "Using Simulation to Evaluate the Batching Approach to Part Type Selection in Flexible Manufacturing Systems," Integrated Manufacturing Systems, vol. 9, no. 1, pp. 5-14, 1998.

[19] M. Denizell and S. Sayin, "Part-Types Selection in Flexible Manufacturings Systems: A Bicriteria Approach with Due Dates," Journal of the Operational Research Society, vol. 49, pp. 659-669, 1998.

[20] A.K. Choudhary, M.K. Tiwari, and J.A. Harding, "Part Selection and Operation-Machine Assignment in a Flexible Manufacturing System Environment: A Genetic Algorithm with Chromosome Differentiation-Based Methodology," Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, vol. 220, no. 5, pp. 677-694, 2006.

[21] S.G. Ponnambalam and L.S. Kiat, "Solving Machine Loading Problem in Flexible Manufacturing Systems Using Particle Swarm Optimization," World Academy of Science, Engineering and Technology, vol. 39, 2008.

[22] D. Pacciarelli. "Loading Parts and Tools in a Flexible Manufacturing System," in Proceedings of the 6th IEEE Mediterranean Conference on Control and Systems. Alghero, Italy, 1998.

[23] R. Swarnkar and M.K. Tiwari, "Modeling Machine Loading Problem of Fmss and Its Solution Methodology Using a Hybrid Tabu Search and Simulated Annealing-Based Heuristic Approach," Robotics and Computer-Integrated Manufacturing, vol. 20, no. 3, pp. 199-209, 2004.

[24] A. Prakash, N. Khilwani, M.K. Tiwari, and Y. Cohen, "Modified Immune Algorithm for Job Selection and Operation Allocation Problem in Flexible Manufacturing Systems," Adv. Eng. Softw., vol. 39, no. 3, pp. 219-232, 2008.

[25] M. Yogeswaran, S.G. Ponnambalam, and M.K. Tiwari, "An Efficient Hybrid Evolutionary Heuristic Using Genetic Algorithm and Simulated Annealing Algorithm to Solve Machine Loading Problem in Fms," International Journal of Production Research, vol. 47, no. 19, pp. 5421-5448, 2009.

[26] S.K. Mukhopadhyay, S. Midha, and V.M. Krishna, "A Heuristic Procedure for Loading Problems in Flexible Manufacturing Systems," International Journal of Production Research, vol. 30, no. 9, pp. 2213, 1992.

[27] K. Seok Shin, J.O. Park, and Y. Keun Kim, "Multi-Objective Fms Process Planning with Various Flexibilities Using a Symbiotic Evolutionary Algorithm," Computers and Operations Research, vol. 38, no. 3, pp. 702-712, 2011.

[28] M. Liang, "Integrating Machining Speed, Part Selection and Machine Loading Decisions in Flexible Manufacturing Systems," Computers & Industrial Engineering, vol. 26, no. 3, pp. 599-608, 1994.

[29] M. Gen and R. Cheng, Genetic Algorithms and Engineering Design, New York: John Wiley & Sons, Inc., 1997.

[30] R.M. Marian, L.H.S. Luong, and K. Abhary, "A Genetic Algorithm for the Optimisation of Assembly Sequences," Comput. Ind. Eng., vol. 50, no. 4, pp. 503-527, 2006.

[31] F. Herrera, M. Lozano, and J.L. Verdegay, "Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis," Artificial Intelligence Review, vol. 12, pp. 265–319, 1998.

[32] R.M. Marian, L. Luong, and S.D. Dao, "Hybrid Genetic Algorithm Optimisation of Distribution Networks—a Comparative Study," in Intelligent Control and Innovative Computing, S.I. Ao, O. Castillo, and X. Huang, Eds., Springer US. p. 109-122, 2012.

[33] N.J. Radcliffe, "Equivalence Class Analysis of Genetic Algorithms," Complex Systems, vol. 5, no. 2, pp. 183–205, 1991.

[34] H. Mühlenbein and D. Schlierkamp-Voosen, "Predictive Models for the Breeder Genetic Algorithm; Continuous Parameter Optimization," Evolutionary Computation vol. 1, pp. 25–49, 1993.