Hybrid Genetic Algorithms for Part Type Selection and Machine Loading Problems with Alternative Production Plans in Flexible Manufacturing System

Wayan F. Mahmudy[1], **Romeo M. Marian**[2], and **Lee H. S. Luong**[3], Non-members

ABSTRACT

This paper addresses two NP-hard and strongly related problems in production planning of flexible manufacturing system (FMS), part type selection problem and machine loading problem. Various flexibilities such as alternative machines, tools, and production plans are considered. Real coded genetic algorithms (RCGA) that uses an array of real numbers as chromosome representation is developed to handle these flexibilities. Hybridizing with variable neighbourhood search (VNS) is performed to improve the power of the RCGA exploring and exploiting the large search space of the problems. The effectiveness of this hybrid genetic algorithm (HGA) is tested using several test bed problems. The HGA improves the FMS effectiveness by considering two objectives, maximizing system throughput and minimizing system unbalance. The resulted objective values are compared to the optimum values produced by branch-and-bound method. The experiments show that the proposed RCGA could reach near optimum solution and the hybridization can improve the performance of the RCGA.

Keywords: Flexible Manufacturing System, Production Planning, Part Type Selection Problem, Machine Loading Problem, Alternative Production Plans, Hybrid Genetic Algorithms

1. INTRODUCTION

Flexible manufacturing system (FMS) is an integrated production system that has capability to manufacture large variety of product in small to medium volume of production batches. Their computer controlled machines can be rapidly configured to produce different products for different market segments. As a high investment is required to acquire FMS, higher resources utilization and maximum system throughput must be achieved to enable early return on investment and keep manufacturers competitive in national and global market. A good production planning is critical to achieve these objectives.

Production planning is done before starting actual production and conducted to ensure that the objectives of the FMS are effectively achieved under several resources limitations. Problems in the production planning stage can be divided into several sub problems such as part type selection problem, machine grouping problem, production ratio problem, resource allocation problem, and machine loading problem [1, 2]. As various different manufacturing environments exist, several combinations of problems in FMS production planning were addressed in literatures. For example, some papers addressed only machine loading problem [3-6] whereas most papers simultaneously solved the part type selection and machine loading problems [7-13]. Another paper simultaneously addressed the part type selection and machine loading problems in the first stage and used the result on this stage to determine the production ratio in the next stage [14]. Moreover, the machine loading problem and the partial machine grouping with tool life constraints was addressed in [15] whereas the part type selection and machine loading problems with tool life constraints was addressed in [16]. This paper focuses on handling various flexibilities when the part type selection and machine loading problems are simultaneously solved.

The flexibility is considered as the main feature of the FMS and can be used to highly utilize all the production resources and at the same time reduce the processing time [17, 18]. The flexibility of FMS refers to an ability to manufacture various products by using same resources (machines and tools). The flexibility of FMS can be divided into two categories, *machine flexibility* and *routing flexibility*, that may be further divided into several sub categories. The machine flexibility refers to possibility reconfiguring machines by attaching different tool types so the machines can perform different operations to produce new type of products [19, 20]. The routing flexibility refers to possibility to manufacture a product through several alternative production routes. Hence, production planning is carried out to fully utilize these flex-

Manuscript received on March 31, 2013 ; revised on June 26, 2013.
Final manuscript received August 19, 2013.
[1] The author is with Department of Computer Science, University of Brawijaya, Indonesia., E-mail: awayanfm@ub.ac.id
[2,3] The authors are with School of Engineering, University of South Australia, Australia., E-mail: romeo.marian@unisa.edu.au and clee.luong@unisa.edu.au

ibilities.

The part type selection problem and the machine loading problem are strongly related problems in production planning of FMS and heavily determine the systems efficiency [19, 21]. The part type selection problem is concerned with decision about which part types (products) on the production order should be taken into a batch to be produced immediately. This decision must be made since the system has several constraints such as certain scheduling period, limited number of machines, limited tool magazines capacity (slots) of each machine and limited number of tools. The machine loading problem deals with allocation of operations for the selected part types and attaching appropriate tool types to the machines [1, 22]. A feasible solution for the machine loading problem may be not obtained if the part type selection problem and the machine problem are addressed separately and sequentially. Thus, solving part type selection and machine loading problems simultaneously is a critical to obtain a feasible solution that provides a higher throughput and maintains the balance of machines workload.

Solving the part type selection and machine loading problems simultaneously requires a powerful method to deal with a large search space. In fact, these production planning problems are considered as NP-hard problems [19] and the optimum solution may not be achieved by exact methods in a reasonable amount of time. Genetic Algorithms (GAs) has been proven as a robust metaheuristic method to solve various problem with a large search space [23]. Our previous research has proven that real coded genetic algorithms (RCGA) could solve the part type selection and machine loading problems and produced near optimum solutions in a reasonable amount of time [8, 9]. Here, flexibilities of operations such as the possibility of operation processed on alternative machines with alternative tools were considered.

This paper as an extension of [24, 25] addresses more complex problem which considers alternative production plans which refer to possibility of producing part on alternative operation sequence. In this paper, the capability of variable neighbourhood search (VNS) is expanded to search the best production plan for the part types. This effort will further exploit the flexibility of the FMS and improve system productivity. Therefore, more powerful method is developed by hybridizing the RCGA with the VNS. Here, the RCGA is designed to explore a large search space of the problems whereas the VNS will improve the power of the RCGA to exploit local areas and obtain better solutions. Hence, the hybrid genetic algorithms (HGA) have a balance power to explore and exploit the search space. A strategy to maintain population diversity is also developed.

2. LITERATURE REVIEW

A number of approaches have been proposed to address the part type selection and machine loading problems such as genetic algorithms [4, 8, 26, 27], particle swarm optimization [19, 28], ant colony optimization [5], immune algorithm [29], multi-agent system [21], symbiotic evolutionary algorithm [30], harmony search algorithm [31], and constraint programming [32]. Hybrid approaches were also developed such as hybridizing genetic algorithm with simulated annealing [22, 33], hybridizing genetic algorithm with particle swarm optimization [12], and hybridizing tabu search with simulated annealing [34]. Even if they reported promising results, not all literatures considered various flexibilities in the FMS due to the complexity of the problems. Here, several simplicities were made to reduce the complexity of the problems [see 8]. This paper attempts to fill this knowledge gap.

As the part type selection and machine loading problems have an important role in determining the productivity and the efficiency of the FMS, an extensive research has been conducted in these areas. Mathematical programming based approaches were applied in few studies. For example, Mgwatu [16] presented two-stage sequential mathematical models as integer nonlinear programming (INLP) problems and used a nonlinear programming software package called LINGO to solved the problems. The part type selection, machine loading and machining optimisation problems were simultaneously solved in the first stage. The scheduling problem was solved in the second stage. Denizell & Sayin [35] formulated the problems as bicriteria mathematical programming problem. Their objective was maximizing system throughput and due-date of part types were used as weight of the objective function. Their model was solved using commercial package software called CPLEX Callable Library12.

Heuristic based approaches were frequently used in recent studies. For instance, Tiwari, Kumar Jha & Bardhan Anand [21] developed a combinatorial auctionbased heuristic for multi-agent system. This approach was used to explore a wide search space of the part type selection and machine loading problems. Biswas & Mahapatra [19] modified particle swarm optimization (PSO) to solve the part type selection and machine loading problems. Their approach 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. Two experimental scenarios were used in their experiments: machine overloading is allowed and is not allowed. Prakash et al. [29] modified immune algorithm to solve the part type selection and machine loading problems. They proposed new hypermutation operator to deal with the drawbacks related to the basic driving forces of the immune algorithms.

Their modified method was claimed more efficient that the original one. The objectives considered were maximizing throughput and maximizing systems balance.

A number of specialised Genetic Algorithms (GAs) were also developed. For example, Abazari, Solimanpur & Sattari [7] developed a GA to address the part type selection and machine loading problems. Infeasible solutions were handled by using penalty value. Yusof, Budiarto & Deris [10] solved the problems using a constraint-chromosome GA. The chromosome representation was designed to produce only feasible solutions and reduce computational time. Each individual had two parts of chromosome, part-sequence and partoperation chromosome. Kumar et al. [36] proposed constraint-based genetic algorithm (CBGA) to handle a complex variety of variables and constraints in the problems. Their CBGA operators were designed to prevent premature convergence by employing exhaustive explorations to exploit the search space.

More complex approaches have been developed by integrating two methods to addresses the part type selection and the machine loading problems. For instance, Arikan & Erol [11] addressed the problems using hybrid simulated annealing (SA) and tabu search algorithm. Infeasible solutions were also handled by using penalty value. Kumar, Murthy & Chandrashekara [12] hybridized a GA and particle swarm optimization. Solution was obtained by converting chromosome using a binary coding system. Tiwari et al. [22] proposed hybrid GA and SA. This approach had a capability to escape from local optimum areas and provide good solutions.

This paper focuses on the developing of chromosome representation for GAs that produces only feasible solutions. The representation can also addresses more complex problem, the existence of alternative production plans which refer to possibility of producing part types on alternative operation sequence. Hybridizing GA with VNS is performed to improve the performance of the GA to exploit local search areas.

3. PROBLEM FORMULATION

This study considers a FMS that is arranged by several computer numerically controlled (CNC) machines and buffers for pre-processed and finished parts. An automated material handling is used to interconnect all machines. Each machine has a tool magazine with certain tool slot capacity. The machines can perform different operations if tooled differently. Several copies of each tool type are available and each copy can be attached to only one machine. Each tool requires a number of slots when it is attached to the machine tool magazine.

When a production order that consists several jobs (part types) arrives, the system selects a set of part types that should be produced immediately as there are a number of technological constraints such as limited number of machines, limited tool magazines capacity (slots) of each machine and limited number of tools. Unselected part types will be manufactured in the next batches.

Each part type can be produced through several alternative production plans. Each production plan requires several machining operations. Each machining operation can be processed on several alternative machines with different tool types and processing time.

Several assumptions are made as follows:

- the production resources such as pallets and fixtures are sufficient;
- machines do not fail during the production period;
- processing times of the operations are deterministic and known in advance;
- machining operation cannot be interrupted.

The problem is formulated as a mixed-integer programming model. However, due to the computational complexity of the problem, solving the problem using mathematical programming based approaches is impractical for large size problems.

3.1 Parameters

The following notations are used in the formulation:

$p=1,\ldots,P$	index for part type.
$a=1,\ldots,A_p$	index for alternative production plan of part type p.
$o=1,\ldots,O_{pa}$	index for operation of production plan a of part type p.
$t=t,\ldots,T$	index for tool type.
$m=1,\ldots,M$	index for machine.
MAG_m	tool magazine capacity on machine m.
N_t	number of tools type t.
S_t	number of slots required by tool type t.
Q_P	batch size (quantity) of part type p.
V_p	value or price of part type p.
MAC_{pao}	set of possible machines on which operation o of production plan a of part type p can be performed.
TM_{paomt}	$=\{1,0\}=1$ if tool type t is required for processing operation o of production plan a of part type p on machine m, and 0 otherwise.
T_{paom}	processing time of operation o of production plan a of part type p on machine m.

3.2 Decision and Depending Variables

The system determines values of several decision variables as follows:

$X_p = \{1, 0\}$ = 1 if part type p is selected in the batch, 0 otherwise.

$X_{pa} = \{1, 0\}$ = 1 if production plan a of part type p is selected, 0 otherwise.

$X_{paom} = \{1, 0\}$ =1 if machine m is chosen to process operation o of production plan a of part type p, 0 otherwise.

The depending variable is variable whose value is determined once the values of the above decision variables are determined. The depending variable for this model is defined as follow:

$Y_{mt} = \{1, 0\}$ 1 if tool type t is loaded to the machine m, and 0 otherwise

3.3 Objectives

Various objectives for the production planning problems have been considered in the literature such as maximizing system throughput, maintaining the balance of the system, minimizing part movement, minimizing tool changeovers, minimizing number of required tools, minimizing machining or production costs, minimizing earliness and tardiness costs, minimizing subcontracting cost of part types, maximizing tool duplication, and minimizing number of batches [30, 33, 37, 38].

Maximizing system throughput and maintaining the balance of the system were frequently used for the production planning of FMS. These objectives can be used to minimize the idle time of the machines that lead to maximal machine utilization and improvement of the overall system output. Maximizing system throughput is used as an objective as there is possibility that not all part types can be produced due to the limited copies of tool types.

Maximizing system throughput is defined as maximizing the value of the selected part types as shown in (1) [29, 33]. If all part types have equal value, the equation calculates the sum of batch size of the selected part types.

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

Maintaining the balance of the system is equal to minimizing system unbalance as expressed in (2) where Wm is workload of machine m. Here, length of scheduling period for each machine (SPm) is determined in advance and overloading of the machines is allowed [19].

$$\text{Minimize} : \sum_{m=1}^{M} |SP_m - W_m| \tag{2}$$

$$\text{where } W_m = \sum_{p=1}^{P} \sum_{a=1}^{A_p} \sum_{o=1}^{O_{pa}} X_{paom} T_{paom} Q_p$$

3.4 Constraints

While considering the objectives, several technological constraints must be satisfied as follows:

– Constraint (3) ensures that one of alternative production plans of the selected part type is chosen.

$$\sum_{a=1}^{Ap} x_{pa} = X_p, \quad p = 1, \ldots, P \tag{3}$$

– Constraint (4) guarantees that all operations of the selected part types are processed.

$$\sum_{o=0}^{O_{pa}} \sum_{m=1}^{M} X_{paom} = O_{pa} X_{pa} \tag{4}$$
$$p = 1, \ldots, P, \quad a = 1, \ldots, A_p$$

– Constraint (5) ensures that each operation of part type is completed on a chosen machine.

$$\sum\nolimits_{m \in MAC_{pao}} X_{paom} = X_p \tag{5}$$
$$p = 1, \ldots, P, \quad a = 1, \ldots, A_p, o = 1, \ldots, O_{pa}$$

– Constraint (6) states that all required tools are loaded to a machine if the machine is selected to process an operation.

$$Y_{mt} = X_{paom} TM_{paom}$$
$$P = 1, \ldots, P, \ a = 1, \ldots, A_p, \ o = 1, \ldots, O_{pa} \tag{6}$$
$$m = 1, \ldots, M, \ t = 1, \ldots, T$$

– Constraint (7) guarantees that the number of tools loaded to the machines do not exceed its availability.

$$\sum_{m=1}^{M} Y_{mi} \leq N_t, \quad t = 1, \ldots, T \tag{7}$$

– Constraint (8) ensures that number of tool slots occupied on a machine magazine must not exceed tool magazine capacity of the machine.

$$\sum_{t=1}^{T} Y_{mt} S_t \leq MAG_m, \quad m = 1, \ldots, M \tag{8}$$

A simple problem set is given to show the complexity of the part type selection and machine loading problems. Table 1 shows different production requirements of seven part types. Apparently, part type 1 has 2 alternative production plans. The first production plan is composed by 2 machining operations whereas the second production plan is composed by

Table 1: *Example of Production Requirement of Part Types.*

part type	batch size	value	prod plan	op	mac	time	tools
1	20	6	1	1	1	20	1 3 4
				2	2	30	4 6
					3	25	2 3
			2	1	2	20	2 3 4
					3	20	3 4 5
				2	1	30	1 3 4
				3	2	20	2 5
					2	30	3 4
2	30	3	1	1	1	20	1 2
				2	2	20	3 4
				3	2	30	5 6 7
3	30	4	1	1	1	30	6 7 8
				2	2	20	1 10
				3	1	20	1 2
			2	1	2	20	2 3 5
				2	3	10	3 6
4	30	2	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	3	1	1	1	50	1 2 3
				2	2	50	4 5
					3	50	4 5 6
			2	1	2	30	1 2
				2	3	30	3 5
6	40	3	1	1	2	20	7 8
				2	2	50	9 10
				3	1	10	3
7	40	4	1	1	2	20	3 4
				2	2	30	1 2
					3	40	8 9

prod plan:production plan, op:operation, mac:machine, time: operation time (seconds)

3 machining operations. Operation 2 of the first production plan has 2 alternative machines, machines 2 or 3. So, part type 1 has total 6 machining routes as follows:

- the first production plan has two machining routes, (1) 1→2 and (2) 1→3;
- the second production plan has four machining routes, (1) 2→1→2, (2) 2→1→2, (3) 3→1→2, and (4) 3→1→2.

Operation 3 of production plan 2 of part type 1 is considered has two alternative machines even it uses same machine (machine 2) since different tools and machining times are required. This flexibility may be considered as *tooling flexibility*.

As there are 7 part types, the part type selection problem deals with 7! part type sequence. For one such ordering there exist: $6\times1\times2\times4\times3\times3\times2 = 5{,}040$ possible machining routes. Thus, for the integrated part type selection and machine loading problems there are $7!\times5{,}040 = 4{,}354{,}560$ possible solutions and the best solution may be found by using complete enumeration or branch-and-bound method. However, the number of possible solutions exponentially increases for larger size problems. Therefore, any proposed approaches must consider this very large search space.

Table 2: *Comparison of ANN, GA, Fuzzy Logic and SOM.*

Tool type	1	2	3	4	5	6	7	8	9	10
copies	2	2	2	2	2	3	3	3	3	3
required slot	1	2	2	1	3	3	4	4	5	5

4. HYBRID GENETIC ALGORITHMS (HGA)

The HGA maintains *pop_size* (population size) of chromosomes in *population* pool to represent possible candidate solutions. Genetic operator (*crossover* and *mutation*) are employed to produce new chromosomes (*offspring*) that are placed in *offspring pool*. A *selection* method is used to determine which chromosomes (from current population and offspring pool) are passed to the next generation. This procedure is repeated until termination condition is achieved. In this study, the iteration is stopped after *tRun* seconds running time. The running time is determined in such way that the HGA reaches convergence cannot obtain better solutions. At the last generation, the best chromosome is decoded as an optimum or a near optimum solution [39]. Fig. 1 shows the cycle of the HGA.

Table 3: *Comparison of ANN, GA, Fuzzy Logic and SOM.*

part type index	1	2	3	4	5	6	7
x	742	220	870	857	846	1126	542
sorted x	220	542	742	846	857	870	1126
part type sequence	2	7	1	5	4	3	6

4.1 Chromosome Representation

The chromosome representation is designed to produces only feasible solutions that minimizes a computational time needed by GAs to move its population toward a feasible search space or repair infeasible chromosomes [40]. Production requirement in Table 1 is used to explain the chromosome construction.

```
Determining parameters of HGA:
    population size pop_size, crossover rate
    cr, mutation rate mr, running time tRun
Initialization: Generate pop_size of random
    chromosomes
WHILE not termination_condition
    Creating new chromosomes
        Produce pop_size×cr offspring by
        using crossover operator and
        pop_size×mr offspring by using
        mutation operator
    Improve offspring using variable
        neighborhood search (VNS)
    Selection
        Perform the next generation by
        selecting pop_size chromosomes from
        parents (population) and offspring
        pool
    Employ population diversity procedure
END WHILE
```

Fig.1: *Pseudo Code of the HGA.*

Table 3 shows an example of chromosome $X = (x_1, x_2, \ldots, x_7)$ that represents 7 part types in Table 1. x_i is treated as a floating number when reproduction operators (crossover and mutation) are employed but it will be rounded to a nearest integer value when decoding operation (into solution) is performed. xi has value in interval $[0, ^{op \times m+p+pp} 2]$. The used variables are detailed as follows:

- *op* is maximum number of operations of part type. Here, *op* is equal to 3 as shown by production plan 1 of part type 1.

- m is number of bits required to represent a binary number in interval [0, maximum number of alternative machines of each operation]. In this case the maximum number of alternative machines of each operation is 2 as shown by operation 2 of production plan 1 of part type 1. Thus, m is equal to 2.

- p is number of bits required to represent a binary number in interval [0, number of part types]. Here, 3 bits are required to represent value in interval [0, 7].

- *pp* is number of bits required to represent a binary number in interval [0, maximum number of production plan of part type]. In this case the maximum number of production plan of part type is 2 as shown in part type 1, 3 and 5. Therefore, *pp* is equal 2.

The smallest position value (SPV) rule is used to get the part types sequence. The rule works by sorting x (together with part type index) in ascending order so a priority of part type that must be produced is obtained as shown in the third row of Table 3. The production plan and machines for operations are obtained by using a binary operation as depicted in Fig. 2. For part type 1, $x_3 = 742$ is converted into $(10\ 1110\ 0110)_2$. Two (according to pp) most right bits $(10)_2$ is used to determining the production plan by using the following formula:

$production plan = (10)_2 \mathrm{mod} pp1 + 1 = 2 \mathrm{mod} 2 + 1 = 1$

mod is modulus operator which produces the remainder of a division and $pp1$ is number of alternative production plans for part type 1. Therefore, production plan 1 is chosen for part type 1.

The next two (according to m) most right bits $(01)_2$ is used to determine the machine for the first operation by using the following formula:

$machine index = (01)_2 \mathrm{mod} np + 1 = 1 \mathrm{mod} 1 + 1 = 1$

np is number of possible machines for operation 1 of production plan 1. Therefore, the first operation of production plan 1 of part type 1 is processed on the first possible machine that is machine 1. By using the next 2 right bits $(10)_2$ and employing the same rule, part type 1 is sequentially processed on machines 1 and 2.

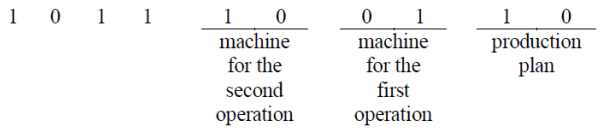

Fig.2: *Binary Operation of Decoding Chromosome.*

The next step of decoding chromosome is loading required tools to the machines. By sequentially choosing part types from the left part of the part type sequence until violates the constraints (tools availability and empty slots on the machines), a batch is performed. For example, after selecting part types 2, 7, 1, 5 and 4 according to the part type sequence as depicted in Table 3, adding part type 3 to the solution violates the constraints. Therefore, the chromosome states that only part types 2, 7, 1, 5 and 4 are selected for the current batch and the objective functions of the problem are calculated based on these selected part types.

The selected part types and their assigned machined is presented in Table 4. Machines workload and their assigned tools are shown in Table 5. Here, there are 3 machines and length of scheduling period for each machine (SP_m) is 4000. Table 6 shows that number of tool types loaded to the machines do not exceed their availability.

Table 4: *Selected Part Types and Their Production Plan and Assigned Machines.*

part type	value	prod plan	machines
2	90	1	1, 2, 2
7	160	1	2, 3
1	120	1	1, 2
5	120	1	1, 2
4	60	1	3, 1, 1
throughput	550		

Table 5: *Selected Part Types and Their Production Plan and Assigned Machines.*

mac	w	unb	slot	used	assigned tools
1	5100	100	15	13	1 2 3 4 6 7
2	4900	100	20	13	3 4 5 6 7
3	2500	2500	25	25	8 9 10
	system unbalance	2700			

mac:machine; w:workload; unb:unbalance
slot:number of slots; used:used/occupied slots

Table 6: *Used Tool Types.*

tool type	1	2	3	4	5	6	7	8	9	10
availability	2	2	2	2	2	3	3	3	3	3
used	1	1	2	2	1	2	2	1	1	1

4.2 Fitness Function

The quality of a chromosome is measured by using a fitness function. Two objectives in (1) and (2) are expressed proportionally and converted to the fitness function in (9). Here, f_1 and f_2 have value between 0 and 1. The weighted parameters w_1 and w_2 can be determined according to the preference of the decision maker.

$$\text{Maximize:} F = w_1 f_1 + w_2 f_2 \tag{9}$$

where:

w_1, w_2: weighted parameters

$$f_1 = \left(\sum_{p=1}^{P} X_p Q_p V_p\right) \Big/ \left(\sum_{p=1}^{P} Q_p V_p\right)$$

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

4.3 Reproduction

Reproduction operators (crossover and mutation) are used to produce new chromosomes during generation. The number of the new chromosomes is determined by the value of crossover rate and mutation rate. Two crossover methods (*flat-crossover* [41] and *extended-intermediate-crossover* [42]) and two mutation methods (*random exchange mutation* and *simple-randommutation*) are used. These methods are effective for the RCGA as proven in our previous research [8, 43].

Fig. 3 gives an example of flat-crossover. $P_1 = (p_1^1, \ldots, p_n^1)$ and $P2 = (p_1^2, \ldots, p_n^2)$ are two selected chromosomes as parents for crossover. Offspring $O = (o_1, \ldots, o_n)$ is produced by generating a random number o_i on interval [pi1,K,pi2]. Fig. 4 gives an example of *extended-intermediate-crossover*. Offspring $O = (o_1, \ldots, o_n)$ is produced by using 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].

The random exchange mutation produces offspring O by choosing two genes randomly from parent P and exchanging their positions as shown in Fig. 5. The simple-random-mutation produces offspring $O = (o_1, \ldots, o_n)$ from parent $P = (p_1, \ldots, p_n)$ by using a formula $o_i = p_i(1 + \alpha_i)$, where α_i is randomly generated on interval [-0.1, 0.1]. The example is given in Fig. 6.

One crossover and one mutation methods are randomly chosen in each generation.

P_1	130	70	92	150	80
P_2	80	65	110	125	83
O	110	68	93	135	82

Fig.3: *Flat-Crossover.*

P_1	130	70	92	150	80
P_2	80	65	110	125	83
α	-0.2	1.2	1.1	-0.1	0.9
O	140	64	111.8	152.5	82.7

Fig.4: *Extended-Intermediate-Crossover.*

	exchange point 1 ↓		exchange point 2 ↓		
P	130	**70**	92	**150**	80
O	130	**150**	92	**70**	80

Fig.5: *Random Exchange Mutation.*

P	130	70	92	150	80
α	0.10	0.06	-0.04	-0.05	-0.08
O	143	74.2	88.32	142.5	73.6

Fig.6: *Simple-Random-Mutation.*

4.4 Selection

During reproduction stage all offspring produced by crossover and mutation are placed on offspring pool. Selection procedure is used to determine which chromosomes from current population and offspring pool are passed to the next generation. Four common selection methods in the literature (roulette wheel, binary tournament, elitist, and replacement) have been examined to determine which method is the most suitable for the RCGA. Here, replacement selection was proved as the best one [9].

The replacement selection has rules as follows:

– Offspring produced by mutation operator will replace their parent if they have better fitness value than their parent.

– Offspring produced by crossover operator (using two parents) will replace their weakest parent if they have better fitness value than their weakest parent.

This replacement selection method guarantees that the best chromosome always passes to the next generation.

4.5 Variable Neighbourhood Search (VNS)

Variable neighborhood search (VNS) is metaheuristic technique that manages a local search (LS) technique. Here, the LS is systematically iterated to explore larger neighborhood until termination condition is achieved. The neighborhood structure is designed to enable the LS exploring the search space from new starting points [44, 45].

As the chromosome representation and reproduction operators of the RCGA are designed to explore a large search space, the VNS is employed to enhance the power of the HGA exploiting local optimum areas. The VNS is employed for each offspring if there is no improvement of the best fitness value on g generations. The proper value of g is determined by conducting preliminary experiments. The neighbourhood structures N_k $(k = 1, \ldots, k_{max})$ is adopted and $N_k(x)$ is defined as the set of solutions in the kth neighbourhood of x.$N_k(x)$ is obtained by randomly changing k production plans of the part types. k_{max} is determined according to the size of problems used in experiments.

A pseudo code for the VNS is shown in Figure 7. Here, the VNS do not change the selected part types in each batch.

The local search works by randomly replacing machine for each operation with other possible machines as shown in Figure 8. If the new solution has better fitness value then it replaces the current solution.

4.6 Maintaining Population Diversity

The performance of GAs is heavily determined by its ability exploring and exploiting the search space. Thus, maintaining the balance of exploration and exploitation of GAs is critical to obtain satisfactory results. For this purpose, this study adopts injecting 20% of new random chromosomes on every 50 generations. By injecting new random chromosomes the population diversity will be maintained. Crossover between the new random chromosomes with current chromosomes in the population will produce offspring that move to other directions in the search space and enable the HGA to escape from local optimum areas.

```
PROCEDURE VariableNeighbourhoodSearch
Input:
      curr:     current solution from the RCGA
      kMax:     number of neighbourhoods
Output:
      best:     the best solution

best ← curr
k ← 1
WHILE k<=kMax DO
   // change k production plans of the part type
   curr ← ChangeProductionPlan (best, k)|
   // find local optimum
   bestLocal ← LocalSearch (curr)
   IF Fitness(bestLocal)> Fitness(best) THEN
      best ← bestLocal
      k ← 1
   ELSE
      k ← k + 1
   END IF
END WHILE
END PROCEDURE
```

Fig.7: *Pseudo Code of The VNS.*

```
PROCEDURE LocalSearch
Input:
      curr:     current solution
Output:
      best:     the best solution

best ← curr
// check each operation of the part type
FOR EACH operation Oi IN curr DO
   //  Change  a  machine  for  Oi  with  other
       possible machine
   curr ← ChangeMachine (curr, Oi)
   IF Fitness(curr)> Fitness(best) THEN
      best ← curr
   END IF
END FOR
END PROCEDURE
```

Fig.8: *Pseudo Code of the Local Search.*

5. RESULT AND DISCUSSION

The HGA is coded in Java and run on personal computer equipped with Intel® Core^{TM} i3-380 processor working at speed 2.53 GHz. Twelve test bed problems with different number of part types are generated as shown in Table 7. Here, problems 1 to 4 represent small size problems, problems 5 to 8 represent medium size problems and problems 5 to 8 represent large size problems. Lengths of scheduling period for all machines are determined in advance and equal within each problem size. Table 8 presents the other randomly generated parameters. The weighted parameters for the fitness function are $w_1 = 1$ and $w_2 = 1$. Several preliminary experiments are carried out to determine appropriate parameter values for the HGA and the results are obtained as follows:

– population size is 100, 200 and 300 for small size problems, medium size problems and large size problems respectively;

– iterations will be stopped after 50, 100, 200 seconds of running time for small size, medium size, and large size problems respectively.

Table 7: *Test-Bed Problems.*

problem	num. of part types	num. of machines	num. of tool types	scheduling period
1	12	4	20	6000
2	12	4	25	6000
3	12	5	20	6000
4	12	5	25	6000
5	24	5	20	9000
6	24	5	25	9000
7	24	6	20	9000
8	24	6	25	9000
9	36	6	20	10000
10	36	6	25	10000
11	36	7	20	10000
12	36	7	25	10000

Table 8: *Randomly Generated Parameters.*

Parameters	Range
tool slot capacity of each machine	40-60
number of copies of each tool type	2-(nMac-1)
number of slots required by each tool	3-7
number of alternative production plans of each part type	1-3
number of operations of each part type	2-(nMac)
batch size of each part type	40-60
value of each part type (dollar)	5-10
number of possible machines for each operation	1-3
processing time of each operation	20-40
number of tool types required for each operation	2-5

nMac: number of machines

Crossover rate and mutation rate must be set in such way that enable the HGA to balance its ability to explore and exploit the search space [46]. Thus, the first stage of experiments is determining the most suitable crossover rate and mutation rate for the HGA. The HGA is run on problem 5 and the crossover rate (cr) is varied from 0 to 0.4. To get a fair comparison, the mutation rate (mr) is set in such way that $cr + mr = 0.4$.

Fig. 9 depicts the average of fitness values from 10 runs for each combination of crossover rate and mutation rate. The best result is achieved at crossover rate of 0.3 and mutation rate of 0.1. Here, by using a low value of crossover rate the HGA will mostly depend on its mutation rate and tend acting as a random search method and cannot learn from previous generations. In other hand, the HGA loses its ability to maintain population diversity if using a high crossover rate and a low mutation rate. By using a high crossover rate the offspring will have a high similarity with their parents and in only few generations the HGA achieves a premature convergence. Here, the HGA losses a chance to explore other areas in the search space and will be trapped in local optimum areas.

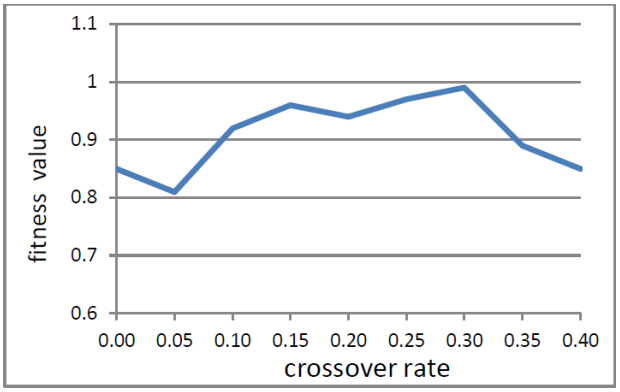

Fig.9: *Average Fitness Values over Different Crossover Rates.*

The experiment is designed to measure the performance of the HGA and also measure the effectiveness of the strategy to maintain population diversity and the hybridization. Therefore, three different approaches are compared in the experiment as the following:

1. The RCGA without the VNS and strategy to maintain population diversity (RCGA_1)

2. The RCGA equipped with the strategy to maintain population diversity (RCGA_2)

3. The RCGA equipped with the VNS and strategy to maintain population diversity (HGA)

The performance of the RCGA_1, the RCGA_2, and the HGA is measured by using deviation of their objective values to the optimum values as shown in (10). F_{opt} is the optimum fitness. FGA_r is fitness value obtained by the approaches in run r. Here, each approach is run 10 times. Branch-and-bound method is used to obtain the optimum solutions. Note that the branch-and-bound method requires average computational time more than 150 hours to solve large size problems that cannot be accepted on daily operation of the FMS.

$$F_{dev} = \left| \frac{\left(F_{opt} - \left(\sum_{r=1}^{10} FGA_r\right)/10\right)}{F_{opt}} \right| \times 100\% \quad (10)$$

The average of throughput and system unbalance from 10 runs is provided in Table 9. In most problems (problems 1, 5, 6, 8, 9, 10, 11, and 12) the HGA produces higher throughput and lower system unbalance comparable to those achieved by the RCGA1 and RCGA2. Lower throughput produced by the HGA as shown in problem 2 is compensated by lower system unbalance. Higher system unbalance produced

Table 9: *Average of Throughput and System Unbalance.*

problem	Optimum Value		$RCGA_1$		$RCGA_2$		HGA	
	TH	SU	TH	SU	TH	SU	TH	SU
1	2,953	2,589	2,550	1,828	2,584	1,909	2,576	1,707
2	2,648	1,202	2,643	1,995	2,652	1,388	2,637	1,265
3	3,526	1,936	3,374	2,312	3,427	2,111	3,471	2,500
4	2,781	2,249	2,532	3,444	2,580	3,361	2,642	3,367
5	3,154	8,281	2,365	14,794	2,426	12,727	2,462	12,714
6	4,509	456	3,953	3,363	4,089	3,105	4,125	2,175
7	3,992	7,645	3,813	14,540	3,602	11,979	3,971	12,427
8	3,581	14,138	2,868	19,064	2,946	18,234	3,065	16,401
9	4,033	13,039	3,433	24,676	3,497	22,758	3,652	22,219
10	4,932	7,708	4,013	14,123	4,015	14,196	4,055	12,767
11	4,900	3,995	3,916	16,421	4,422	13,320	4,300	12,484
12	5,815	8,352	4,617	15,821	4,760	12,096	4,864	11,636

Table 10: *Comparison of Average of Fitness Values.*

problem	F_{opt}	$RCGA_1$			$RCGA_2$			HGA		
		F	F_{dev}	itr best	F	F_{dev}	itr best	F	F_{dev}	itr best
1	1.575	1.514	3.9	4,396	1.518	3.6	8,278	1.525	3.2	9,016
2	1.477	1.443	2.3	2,019	1.471	0.5	8,375	1.473	0.3	8,146
3	1.638	1.595	2.6	5,632	1.613	1.6	6,803	1.609	1.8	7,660
4	1.542	1.447	6.2	7,026	1.460	5.3	10,593	1.474	4.4	7,394
5	1.175	0.940	20.0	2,525	0.993	15.5	4,948	0.998	15.1	7,566
6	1.483	1.358	8.5	145	1.378	7.1	5,252	1.403	5.4	8,671
7	1.321	1.172	11.2	3,639	1.195	9.5	5,357	1.230	6.9	6,144
8	1.134	0.964	15.0	3,152	0.988	12.9	6,531	1.035	8.7	2,357
9	1.063	0.828	22.2	154	0.864	18.7	2,389	0.884	16.9	4,929
10	1.219	1.048	14.1	2,024	1.047	14.2	8,086	1.073	12.0	6,610
11	1.318	1.065	19.2	1,917	1.148	12.9	3,887	1.151	12.7	5,365
12	1.274	1.086	14.7	140	1.149	9.8	4,882	1.163	8.7	7,044

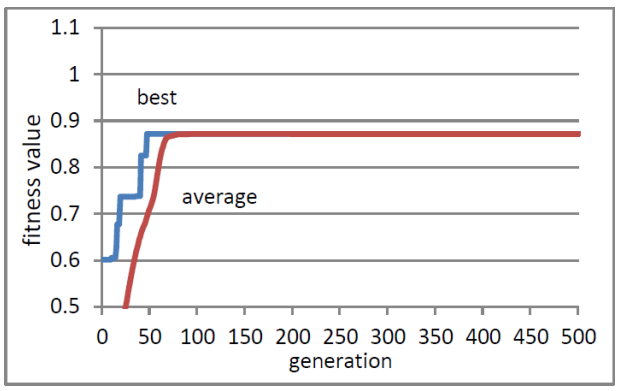

Fig.10: *The Best and Average Fitness Value Resulted by the $RCGA_1$.*

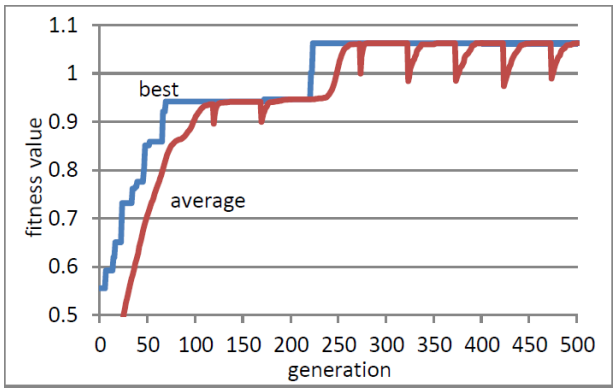

Fig.11: *The Best and Average Fitness Value Resulted by the HGA.*

by the HGA as shown in problems 3, 4, and 7 is compensated by higher throughput.

Table 10 shows that the HGA consistently achieves better results (higher fitness value) in all test bed problems comparable to those achieved by the $RCGA_1$ and the $RCGA_2$. Thus, it proves the effectiveness of the hybridization. The effectiveness of the strategy to maintain population diversity is also proved. Here, the $RCGA_2$ outperforms the $RCGA_1$ in 11 out 12 problems. The $RCGA_1$ is slightly better than the $RCGA_2$ in only problem 10.

Table 10 also reveals that all approaches tend to produce higher F_{dev} on larger size problem. On all small size problems, the HGA achieves F_{dev} below 5%. On medium size problems the worst solution achieved by the HGA is on problem 8 with F_{dev} of 15.1% whereas in the large size problems the worst solution is on problem 9 with F_{dev} of 16.9%. The average of F_{dev} in the large size problems is 12.6%. This result could be considered as a good result as the running time for large size problem is only 200 seconds. It is should be noted that the purpose of the experiments is to prove the effectiveness of the strategy to maintain population diversity and the hybridization. In real manufacturing environment a better result may be achieved by increasing population size and running time of the HGA.

The effectiveness of the HGA is also shown by number of iterations to obtain the best solution (*itr* best) in Table 10. The HGA have significantly higher *itr* best than the $RCGA_1$ and the $RCGA_2$. Here, the $RCGA_1$ and the $RCGA_2$ achieve their convergence faster which may indicate that they are trapped in local optimum areas and cannot obtain a better solution.

To show the difference of behaviour of the RCGA1 and the HGA during generation, both approaches are run on problem 5. The best and average of fitness values are presented in Fig. 10 and Fig. 11. Fig. 10 reveals that the RCGA1 experience early convergence and cannot obtain better result after 69 generations. In contrast, Fig. 11 shows the effect of the VNS and the strategy to maintain population diversity to the average of fitness values produced by the HGA. The strategy to maintain population diversity causes a fluctuation on the average of fitness values as new chromosomes that may have lower fitness values are injected to the population. Recombination between the new random chromosomes with current chromosomes in the population produce offspring that explore other directions in the search space and enable the HGA to escape from local optimum areas. It is indicated by its higher number of iterations to obtain the best solution. Here, the HGA achieves convergence after 2795 generations.

6. CONCLUSIONS

This paper presents the development of a model for the optimization of the integrated part type selection and machine loading problems. The tool allocation problem is considered as the integral part of the machine loading problem. The chromosome representation of the RCGA is designed to address various flexibilities of operations in the FMS. Here, the flexibility of the FMS on realistic manufacture environment is exploited to improve system efficiency and productivity. The experiment proves that hybridizing the RCGA with variable neighbourhood search (VNS) and a strategy to maintain population diversity is effective to optimize the objective of the system in all test bed problems.

Our further work will focus on the integration of production planning and scheduling in FMS. A robust and efficient method is required to address this complex problem. Combining the HGA with Multi Agent System (MAS) will be considered.

References

[1] K. E. Stecke, "Design, planning, scheduling, and control problems of flexible manufacturing systems," *Annals of Operations Research*, vol. 3, pp. 1–12, 1985.

[2] S. Özpeynirci and M. Azizoglu, "Bounding approaches for operation assignment and capacity allocation problem in flexible manufacturing systems," *Computers & Operations Research*, vol. 36, pp. 2531–2540, 2009.

[3] Y.-D. Kim, G.-C. Lee, S.-K. Lim, and S.-K. Choi, "Tool requirements planning in a flexible manufacturing system: minimizing tool costs subject to a makespan constraint," *International Journal of Production Research*, vol. 41, pp. 3339–3357, 2003.

[4] J.-H. Chen and S.-Y. Ho, "A novel approach to production planning of flexible manufacturing systems using an efficient multi-objective genetic algorithm," *International Journal of Machine Tools and Manufacture*, vol. 45, pp. 949–957, 2005.

[5] F. T. S. Chan and R. Swarnkar, "Ant colony optimization approach to a fuzzy goal programming model for a machine tool selection and operation allocation problem in an FMS," *Robotics and Computer-Integrated Manufacturing*, vol. 22, pp. 353–362, 2006.

[6] S. Bilgin and M. Azizoglu, "Capacity and tool allocation problem in flexible manufacturing systems," *The Journal of the Operational Research Society*, vol. 57, pp. 670–681, 2006.

[7] A. M. Abazari, M. Solimanpur, and H. Sattari, "Optimum loading of machines in a flexible manufacturing system using a mixed-integer linear mathematical programming model and genetic

algorithm," *Computers & Industrial Engineering*, vol. 62, pp. 469–478, 2012.

[8] W. F. Mahmudy, R. M. Marian, and L. H. S. Luong, "Solving part type selection and loading problem in flexible manufacturing system using real coded genetic algorithms Part I:modeling," *World Academy of Science, Engineering and Technology*, vol. 69, pp. 773–779, 2012.

[9] W. F. Mahmudy, R. M. Marian, and L. H. S. Luong,"Solving part type selection and loading problem in flexible manufacturing system using real coded genetic algorithms Part II: optimization," *World Academy of Science, Engineering and Technology*, vol. 69, pp. 778–782, 2012.

[10] U. K. Yusof, R. Budiarto, and S. Deris, "Constraint-chromosome genetic algorithm for flexible manufacturing system machine-loading problem," *International Journal of Innovative Computing, Information and Control*, vol. 8, pp.1591–1609, 2012.

[11] M. Arikan and S. Erol, "A hybrid simulated annealing-tabu search algorithm for the part selection and machine loading problems in flexible manufacturing systems," *The International Journal of Advanced Manufacturing Technology*, vol. 59, pp. 669–679, 2012/03/01 2012.

[12] V. M. Kumar, A. N. N. Murthy, and K. Chandrashekara, "A hybrid algorithm optimization approach for machine loading problem in flexible manufacturing system," *Journal of Industrial Engineering International*, vol. 8, pp. 1–10, 2012.

[13] M. Goswami and M. K. Tiwari, "A reallocation-based heuristic to solve a machine loading problem with material handling constraint in a flexible manufacturing system," *International Journal of Production Research*, vol. 44, pp.569–588, 2006.

[14] 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, pp. 5–14, 1998.

[15] H.-W. Kim, J.-M. Yu, J.-S. Kim, H.-H. Doh, D.- H. Lee, and S.-H. Nam, "Loading algorithms for flexible manufacturing systems with partially grouped unrelated machines and additional tooling constraints," *The International Journal of Advanced Manufacturing Technology*, vol. 58, pp. 683–691, 2012.

[16] M. I. Mgwatu, "Integration of part selection, machine loading and machining optimisation decisions for balanced workload in flexible manufacturing system," *International Journal of Industrial Engineering Computations*, vol. 2, pp.913-930, 2011.

[17] H. T. N. I. K. Nejad, N. Sugimura, K. Iwamura, and Y. Tanimizu, "Integrated dynamic process planning and scheduling in flexible manufacturing systems via autonomous agents," *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, vol. 2, pp. 719–734, 2008.

[18] A. H. R. Zaied, "Quantitative models for planning and scheduling of flexible manufacturing system," *Emirates Journal for Engineering Research*, vol. 13, pp. 11–19, 2008.

[19] 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, pp. 931–942, 2008.

[20] 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, pp. 87– 102, 2004.

[21] 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, pp. 312-324, 2010.

[22] 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 CyberneticsPart A: Systems and Humans*, vol. 36, pp. 1170–1184, 2006.

[23] W. Shen, "Genetic algorithms in agent-based manufacturing scheduling systems," *Integr. Comput.-Aided Eng.*, vol. 9, pp. 207–217, 2002.

[24] W. F. Mahmudy, R. M. Marian, and L. H. S. Luong, "Optimization of part type selection and loading problem with alternative production plans in flexible manufacturing system using hybrid genetic algorithms Part 1: modelling and representation," in *5th International Conference on Knowledge and Smart Technology (KST)*, Chonburi, Thailand, 2013, pp. 75–80.

[25] W. F. Mahmudy, R. M. Marian, and L. H. S. Luong, "Optimization of part type selection and loading problem with alternative production plans in flexible manufacturing system using hybrid genetic algorithms Part 2: genetic operators & results," in *5th International Conference on Knowledge and Smart Technology (KST)*, Chonburi, Thailand, 2013, pp. 81–85.

[26] 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 Manufac-*

ture, vol. 220, pp. 677– 694, May 1 2006.

[27] N. Kumar and K. Shanker, "A genetic algorithm for FMS part type selection and machine loading," *International Journal of Production Research*, vol. 38, pp. 3861–3887, 2000.

[28] 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.

[29] 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, pp. 219–232, 2008.

[30] 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, pp. 702–712, 2011.

[31] U. K. Yusof, R. Budiarto, and S. Deris, "Harmony search algorithm for flexible manufacturing system(FMS) machine loading problem," *presented at the 2011 3rd Conference on Data Mining and Optimization (DMO)*, Selangor Malaysia, 2011.

[32] L. J. Zeballos, "A constraint programming approach to tool allocation and production scheduling in flexible manufacturing systems," *Robotics and Computer-Integrated Manufacturing*, vol. 26, pp. 725–743, 2010.

[33] 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, pp. 5421–5448, 2009.

[34] 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, pp. 199–209, 2004.

[35] 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.

[36] A. Kumar, Prakash, M. K. Tiwari, R. Shankar, A. Baveja, and D.B.Shanmugam, "Solving machine-loading problem of a flexible manufacturing system with constraint-based genetic algorithm," *European Journal of Operational Research*, vol. 175, pp. 1043-1069, 2006.

[37] G. K. Nayak and D. Acharya, "Part type selection, machine loading and part type volume determination problems in FMS planning," *International Journal of Production Research*, vol. 36, pp. 1801–1824, 1998/07/01 1998.

[38] D. H. Lee and Y. D. Kim, "Iterative procedures for multi-period order selection and loading problems in flexible manufacturing system," *Int J Prod Res*, vol. 36, pp. 2653-2668, 1998.

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

[40] R. M. Marian, L. Luong, and S. D. Dao, "Hybrid genetic algorithm optimisation of distribution networksa comparative study," in *Intelligent Control and Innovative Computing*, vol. 110, S. I. Ao, O. Castillo, and X. Huang, Eds., ed US:Springer, 2012, pp. 109–122.

[41] N. J. Radcliffe, "Equivalence class analysis of genetic algorithms," *Complex Systems*, vol. 5, pp. 183-205, 1991.

[42] H. Muhlenbein and D. Schlierkamp-Voosen, "Predictive models for the breeder genetic algorithm; continuous parameter optimization," *Evolutionary Computation*, vol. 1, pp. 25-49, 1993.

[43] W. F. Mahmudy, R. M. Marian, and L. H. S. Luong, "Real coded genetic algorithms for solving flexible job-shop scheduling problem Part II: optimization," *Advanced Materials Research*, vol. 701, pp. 364–369, 2013.

[44] P. Hansen and N. Mladenovi, "An introduction to variable neighborhood search," in *Metaheuristics*, ed: Springer US, 1999, pp. 433–458.

[45] P. Hansen and N. Mladenovi, "Variable neighborhood search: Principles and applications," *European Journal of Operational Research*, vol. 130, pp. 449–467, 2001.

[46] M. Lozano and F. Herrera, "Fuzzy adaptive genetic algorithms: design, taxonomy, and future directions," *Soft Computing*, vol. 7, pp. 545-562, 2003.

Wayan F. Mahmudy obtained bachelor degree in mathematics from University of Brawijaya, Indonesia, master degree in information technology from Institut Teknologi Sepuluh November (ITS), Indonesia, and completed his Ph.D. in Manufacturing Engineering at University of South Australia. He is a Lecturer at Department of Computer Science, University of Brawijaya (UB), Indonesia. His research interests include optimization of combinatorial problems and machine learning.

Romeo M. Marian is a Senior Lecturer and Program Director in the School of Engineering at University of South Australia (UniSA). He graduated from the Technical University of Cluj-Napoca, Romania, with a BE (Hons) in Mechanical Mechatronics Engineering, specializing in design of Machine-Tools and Robots. He completed a Master of Science, organized under a Tempus Programme of the European Union by a cluster of European Universities. His doctoral studies, at UniSA, concerned the modeling and optimization of assembly operations using Genetic Algorithms.

Lee H. S. Luong is a Professor of Manufacturing Engineering in the School of Engineering at University of South Australia (UniSA). He received BEng.(Hons) and Ph.D. degrees from Monash University. His research interests are in the areas of logistics and supply chain management (distribution network design, inventory management, ERP, planning and scheduling), sustainable product design and manufacture (design for dis-assembly, design for recyclability and product upgrade, close-loop supply chain, LCA and environmental impact assessment), and cellular and flexible manufacturing systems.