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

Wayan F. Mahmudy1, Romeo M. Marian2, and Lee H. S. Luong3, 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-

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
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 optimization 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 auction-based 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 part-operation 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 & Chandrashekar [12] hybridized a GA and particle swarm optimization. Solution was obtained by converting the 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:

\[ p = 1, \ldots, P \]  \quad \text{index for part type.}
\[ a = 1, \ldots, A_p \]  \quad \text{index for alternative production plan of part type } p.
\[ o = 1, \ldots, O_{pa} \]  \quad \text{index for operation of production plan } a \text{ of part type } p.
\[ t = t_1, \ldots, T \]  \quad \text{index for tool type.}
\[ m = 1, \ldots, M \]  \quad \text{index for machine.}
\[ MAG_m \]  \quad \text{tool magazine capacity on machine } m.
\[ N_t \]  \quad \text{number of tools type } t.
\[ S_t \]  \quad \text{number of slots required by tool type } t.
\[ Q_p \]  \quad \text{batch size (quantity) of part type } p.
\[ V_p \]  \quad \text{value or price of part type } p.
\[ MAC_{pao} \]  \quad \text{set of possible machines on which operation } o \text{ of production plan } a \text{ of part type } p \text{ can be performed.}
\[ TM_{paom} = \{1,0\} = 1 \text{ if tool type } t \text{ is required for processing operation } o \text{ of production plan } a \text{ of part type } p \text{ on machine } m, \text{ and } 0 \text{ otherwise.}
\[ T_{paom} \]  \quad \text{processing time of operation } o \text{ of production plan } a \text{ of part type } p \text{ 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 changes, 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.

\[
\text{Maximize: } \sum_{p=1}^{P} X_p Q_p V_p \quad (1)
\]

Maintaining the balance of the system is equal to minimizing system unbalance as expressed in (2) where W_m 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].

\[
\text{Minimize: } \sum_{m=1}^{M} |SP_m - W_m| \quad (2)
\]

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.

\[
\sum_{a=1}^{A_p} x_{pa} = X_p, \; p = 1, \ldots, P \quad (3)
\]

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

\[
\sum_{o=0}^{O_{pa}} \sum_{m=1}^{M} X_{paom} = O_{pa} x_{pa} \quad (4)
\]

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

\[
\sum_{m \in MAC_{pa}} X_{paom} = X_p \quad (5)
\]

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

\[
Y_{mt} = X_{paom} T_{paom} \quad (6)
\]

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

\[
\sum_{m=1}^{M} Y_{mt} \leq N_t, \; t = 1, \ldots, T \quad (7)
\]

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

\[
\sum_{t=1}^{T} Y_{mt} S_t \leq MAC_m, \; m = 1, \ldots, M \quad (8)
\]

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.

<table>
<thead>
<tr>
<th>part type</th>
<th>batch size</th>
<th>value</th>
<th>prod plan</th>
<th>op</th>
<th>mac</th>
<th>time</th>
<th>tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>20</td>
<td>1 3 4</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>30</td>
<td>4</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>25</td>
<td>2</td>
<td>2</td>
<td>20</td>
<td>3</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>30</td>
<td>3</td>
<td>20</td>
<td>3</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>30</td>
<td>1</td>
<td>1</td>
<td>30</td>
<td>1</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>30</td>
<td>2</td>
<td>20</td>
<td>3</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>30</td>
<td>3</td>
<td>30</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Tool type</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>copies</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>required slot</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>part type index</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>742</td>
<td>220</td>
<td>870</td>
<td>857</td>
<td>846</td>
<td>1126</td>
<td>542</td>
</tr>
<tr>
<td>sorted x</td>
<td>220</td>
<td>542</td>
<td>742</td>
<td>846</td>
<td>857</td>
<td>870</td>
<td>1126</td>
</tr>
</tbody>
</table>

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.
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. $x_i$ has value in interval $[0, \text{op}_m + \text{pp}]$. The used variables are detailed as follows:

- $\text{op}$ is maximum number of operations of part type. Here, $\text{op}$ 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 $[0, \text{maximum number of alternative machines of each operation}]$. 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 $[0, \text{number of part types}]$. Here, 3 bits are required to represent value in interval $[0, 7]$.
- $\text{pp}$ is number of bits required to represent a binary number in interval $[0, \text{maximum number of production plan of part type}]$. In this case the maximum number of production plan of part type is 2 as shown in part type 1, 3 and 5. Therefore, $\text{pp}$ 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 $(101100110)_2$. Two (according to pp) most right bits $(10)$ are used to determine the production plan by using the following formula:

$$\text{production plan} = (10)_2 \text{mod} \text{pp} + 1 = 2 \text{mod} 2 + 1 = 1$$

$\text{mod}$ is modulus operator which produces the remainder of a division and $\text{pp}$ 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 $(01)_2$ are used to determine the machine for the first operation by using the following formula:

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

$\text{np}$ 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.

Table 4 shows selected part types and their production plan and assigned machines. 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 5: Selected Part Types and Their Production Plan and Assigned Machines.

<table>
<thead>
<tr>
<th>mac</th>
<th>w</th>
<th>umb</th>
<th>slot</th>
<th>used</th>
<th>assigned tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5100</td>
<td>100</td>
<td>15</td>
<td>13</td>
<td>1 2 3 4 6 7</td>
</tr>
<tr>
<td>2</td>
<td>4900</td>
<td>100</td>
<td>20</td>
<td>13</td>
<td>3 4 5 6 7</td>
</tr>
<tr>
<td>3</td>
<td>2500</td>
<td>2500</td>
<td>25</td>
<td>25</td>
<td>8 9 10</td>
</tr>
</tbody>
</table>

system 2700unbalance
mac:machine; w:workload; unb:unbalance
slot:number of slots; used:used/occupied slots

Table 6: Used Tool Types.

<table>
<thead>
<tr>
<th>tool type</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>availability</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>used</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

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.

Maximize:$F = w_1 f_1 + w_2 f_2$  
(9)

where:

$w_1, w_2$: weighted parameters

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

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

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 $P_2 = (p_1^2, \ldots, p_n^2)$ 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 $[p_i^1, K, p_i^2]$. 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 $\alpha_i$ 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 $\alpha_i$ 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.

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 a 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 neighborhood structures \( N_k(x) \) is adopted and \( N_k(x) \) is defined as the set of solutions in the kth neighborhood of \( x \). \( N_k(x) \) is obtained by randomly changing \( k \) production plans of the part types. \( k_{\text{max}} \) 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.

5. RESULT AND DISCUSSION

The HGA is coded in Java and run on personal computer equipped with Intel® Core™ 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 9 to 12 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.

<table>
<thead>
<tr>
<th>problem</th>
<th>num. of part types</th>
<th>num. of machines</th>
<th>num. of tool types</th>
<th>scheduling period</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12</td>
<td>4</td>
<td>20</td>
<td>6000</td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td>4</td>
<td>25</td>
<td>6000</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>5</td>
<td>20</td>
<td>6000</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>5</td>
<td>25</td>
<td>9000</td>
</tr>
<tr>
<td>5</td>
<td>24</td>
<td>5</td>
<td>20</td>
<td>9000</td>
</tr>
<tr>
<td>6</td>
<td>24</td>
<td>5</td>
<td>25</td>
<td>9000</td>
</tr>
<tr>
<td>7</td>
<td>24</td>
<td>6</td>
<td>20</td>
<td>9000</td>
</tr>
<tr>
<td>8</td>
<td>24</td>
<td>6</td>
<td>25</td>
<td>9000</td>
</tr>
<tr>
<td>9</td>
<td>36</td>
<td>6</td>
<td>20</td>
<td>10000</td>
</tr>
<tr>
<td>10</td>
<td>36</td>
<td>6</td>
<td>25</td>
<td>10000</td>
</tr>
<tr>
<td>11</td>
<td>36</td>
<td>7</td>
<td>20</td>
<td>10000</td>
</tr>
<tr>
<td>12</td>
<td>36</td>
<td>7</td>
<td>25</td>
<td>10000</td>
</tr>
</tbody>
</table>

Table 8: Randomly Generated Parameters.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>tool slot capacity of each machine</td>
<td>40-60</td>
</tr>
<tr>
<td>number of copies of each tool type</td>
<td>2-(n\text{Mac}-1)</td>
</tr>
<tr>
<td>number of slots required by each tool</td>
<td>3-7</td>
</tr>
<tr>
<td>number of alternative production plans</td>
<td>1-3</td>
</tr>
<tr>
<td>of each part type</td>
<td></td>
</tr>
<tr>
<td>number of operations of each part type</td>
<td>2-(n\text{Mac})</td>
</tr>
<tr>
<td>batch size of each part type</td>
<td>40-60</td>
</tr>
<tr>
<td>value of each part type (dollar)</td>
<td>5-10</td>
</tr>
<tr>
<td>number of possible machines for each</td>
<td>1-3</td>
</tr>
<tr>
<td>operation</td>
<td></td>
</tr>
<tr>
<td>processing time of each operation</td>
<td>20-40</td>
</tr>
<tr>
<td>number of tool types required for each</td>
<td>2-5</td>
</tr>
<tr>
<td>operation</td>
<td></td>
</tr>
</tbody>
</table>

\(n\text{Mac}: \text{number of machines}\)

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 (RCGA1)
2. The RCGA equipped with the strategy to maintain population diversity (RCGA2)
3. The RCGA equipped with the VNS and strategy to maintain population diversity (HGA)

The performance of the RCGA1, the RCGA2, 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.

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

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.

<table>
<thead>
<tr>
<th>problem</th>
<th>Optimum Value</th>
<th>RCGA₁</th>
<th>RCGA₂</th>
<th>HGA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TH</td>
<td>SU</td>
<td>TH</td>
<td>SU</td>
</tr>
<tr>
<td>1</td>
<td>2,953</td>
<td>2,589</td>
<td>2,550</td>
<td>1,828</td>
</tr>
<tr>
<td>2</td>
<td>2,648</td>
<td>1,202</td>
<td>2,643</td>
<td>1,995</td>
</tr>
<tr>
<td>3</td>
<td>3,526</td>
<td>1,936</td>
<td>3,374</td>
<td>2,312</td>
</tr>
<tr>
<td>4</td>
<td>2,781</td>
<td>2,249</td>
<td>2,532</td>
<td>3,444</td>
</tr>
<tr>
<td>5</td>
<td>3,154</td>
<td>8,281</td>
<td>2,365</td>
<td>14,794</td>
</tr>
<tr>
<td>6</td>
<td>4,509</td>
<td>456</td>
<td>3,953</td>
<td>3,363</td>
</tr>
<tr>
<td>7</td>
<td>3,992</td>
<td>7,645</td>
<td>3,813</td>
<td>14,540</td>
</tr>
<tr>
<td>8</td>
<td>3,581</td>
<td>14,138</td>
<td>2,868</td>
<td>19,064</td>
</tr>
<tr>
<td>9</td>
<td>4,033</td>
<td>13,039</td>
<td>3,433</td>
<td>24,676</td>
</tr>
<tr>
<td>10</td>
<td>4,932</td>
<td>7,708</td>
<td>4,013</td>
<td>14,123</td>
</tr>
<tr>
<td>11</td>
<td>4,900</td>
<td>3,995</td>
<td>3,916</td>
<td>16,421</td>
</tr>
<tr>
<td>12</td>
<td>5,815</td>
<td>8,352</td>
<td>4,617</td>
<td>15,821</td>
</tr>
</tbody>
</table>

Table 10: Comparison of Average of Fitness Values.

<table>
<thead>
<tr>
<th>problem</th>
<th>$F_{opt}$</th>
<th>RCGA₁</th>
<th>RCGA₂</th>
<th>HGA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$F$</td>
<td>$F_{ave}$</td>
<td>it. best</td>
<td>$F$</td>
</tr>
<tr>
<td>1</td>
<td>1.575</td>
<td>1.514</td>
<td>3.9</td>
<td>4.396</td>
</tr>
<tr>
<td>2</td>
<td>1.477</td>
<td>1.443</td>
<td>2.3</td>
<td>2.019</td>
</tr>
<tr>
<td>3</td>
<td>1.638</td>
<td>1.595</td>
<td>2.6</td>
<td>5.632</td>
</tr>
<tr>
<td>4</td>
<td>1.542</td>
<td>1.447</td>
<td>6.2</td>
<td>7.026</td>
</tr>
<tr>
<td>5</td>
<td>1.175</td>
<td>0.940</td>
<td>20.0</td>
<td>2.525</td>
</tr>
<tr>
<td>6</td>
<td>1.483</td>
<td>1.358</td>
<td>8.5</td>
<td>145</td>
</tr>
<tr>
<td>7</td>
<td>1.321</td>
<td>1.172</td>
<td>11.2</td>
<td>3.639</td>
</tr>
<tr>
<td>8</td>
<td>1.134</td>
<td>0.964</td>
<td>15.0</td>
<td>3,152</td>
</tr>
<tr>
<td>9</td>
<td>1.063</td>
<td>0.828</td>
<td>22.2</td>
<td>154</td>
</tr>
<tr>
<td>10</td>
<td>1.219</td>
<td>1.048</td>
<td>14.1</td>
<td>2,024</td>
</tr>
<tr>
<td>11</td>
<td>1.318</td>
<td>1.065</td>
<td>19.2</td>
<td>1,917</td>
</tr>
<tr>
<td>12</td>
<td>1.274</td>
<td>1.086</td>
<td>14.7</td>
<td>140</td>
</tr>
</tbody>
</table>

Fig.10: The Best and Average Fitness Value Resulted by the RCGA₁.

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 RCGA1 and the RCGA2. Thus, it proves the effectiveness of the hybridization. The effectiveness of the strategy to maintain population diversity is also proved. Here, the RCGA2 outperforms the RCGA1 in 11 out 12 problems. The RCGA1 is slightly better than the RCGA2 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 RCGA1 and the RCGA2. Here, the RCGA1 and the RCGA2 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


Wayan F. Mahmudy obtained bachelor degree in mathematics from University of Brawijaya, Indonesia, master degree in information technology from Institut Teknologi Sepuluh Novem ber (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.