Abstract  Maintenance is important to manufacturing process with a view to sustaining production efficiency. In this study scheduling of unrelated parallel machines with effects of learning and aging and multi maintenance activities is considered. The objective is to find the optimal maintenance periods, optimal maintenance frequencies and optimal job sequences such that the makespan of all jobs are minimized. Polynomial algorithm is provided and two heuristic algorithms are presented. Computational experiments are conducted to evaluate their performances.

Keywords  Unrelated Parallel Machines, Learning Effect, Aging Effect, Maintenance Activities, Makespan

1. Introduction

Traditionally, the processing time of jobs is assumed to be fixed and constant values. In practice the actual processing time of a job maybe subject to change due to effects of learning and aging. Gupta and Gupta [14] were the first introduce deteriorating jobs in the scheduling problems. Mosheiov [13] considered the single machine problem under linear deterioration for minimizing the total completion time where the processing time of the jobs are the same. It shows that the optimal schedule is V-shaped. Lee [16] was the pioneer to introduce the learning and deterioration effects simultaneously in scheduling problems.Cheng, et al [9] investigated a new scheduling model with learning effect deterioration jobs and setup times. They proposed an optimal solution for single-machine problem in polynomial-time. Wang and Wang [8] provided resource allocation scheduling with considering learning effects, where the job’s processing time is a function of the job’s position in a sequence and its resource allocation. They suggested a polynomial-time algorithm to solve the optimal job sequence and resource allocation. Zhang and Yan [7] studied a learning effect model and find optimal solution for single machine and flowshop problem .Lee and Lai [6] explored a model with effects of learning and deterioration which the processing time of a job is a general function of the normal processing times of the jobs already processed and its scheduled positions.

In a real manufacturing process, machine may not be available due to preventive maintenance, periodic repairs, tools changes or breakdown so a machine may require multi-maintenance activities to improve its production efficiency or product quality. The problem of joint scheduling and maintenance has become a popular topic and has received the attention of many researchers. The reader can refer to Schmidt [17] and Ma et-al [18]. Kuo and Yang [3] considered single-machine scheduling problems with aging effects and multi-maintenance activities in acyclic process. They studied the problem with job independent and position dependent aging effect to minimize the makespan and propose an algorithm to solve the problem in polynomial time. Zhao and Tang [4] extended the investigation of Kuo and Yang [3] in addition to job dependent aging effect. They solved the problem in polynomial-time. Yang and Yang [5] considered single machine scheduling problem with the deterioration effect under a deterioration maintenance activity consideration. They suggested an algorithm to solve in polynomial time. Toksari and Guner [10] considered a parallel machine earliness and tardiness scheduling problem with the effect of learning and deterioration under different penalties. Chen [20]studied a single machine scheduling problem to minimize the number of tardy jobs under periodic maintenance activities and non-resumable jobs. He investigated a heuristic algorithm to find near-optimal solutions and provided a branch and bound algorithm to solve the problem. Okoliwski and Gawiejnowiz [11] studied the parallel machine problem that learning effect on job processing time is modeled by the general Dejong’s learning curve to minimize the makespan. Hsu, Kuo and Yang [12] considered an unrelated parallel machine problem with learning effect and setup time that the setup time of each job is dependent to past sequence. They proposed a solution in polynomial time to minimize the total completion time.

In this study, unrelated parallel machine scheduling problems with the effects of aging and learning and multi maintenance activities is studied. The objective is to
minimize the makespan. The remaining of paper is organized as follows: in section 2 the problem is described and formulated. Two polynomial algorithms are presented to solve the problem under study in section 3. In section 4 the numerical examples are solved and finally calculations are given in section 5.

2. Problem Definition

In this section, the notation that is used throughout the paper will be explained first, followed by the formulation of the problem.

\( n \) The total number of jobs;
\( m \) The total number of machines;
\( t_h \) The duration of each maintenance activity, \( h = 1, 2, \ldots, m \);
\( C_j \) The completion time of job \( j \);

It is considered that there are \( n \) jobs \( j = \{ j_1, j_2, \ldots, j_n \} \) to be processed on \( m \) unrelated parallel machines \( (M_h, h = 1, 2, \ldots, m) \). All the jobs are simultaneously available at time zero and the job preemption is not allowed. The machines can process at most one job at a time. To counteract the aging effect, the machines may require maintenance activities to improve its production efficiency, during the maintenance, the machine is stopped.

We formulated the problem under study as assigned a maintenance activity in the first position on each machine. We assumed that the first maintenance activity duration is zero. All the jobs are simultaneously available at time zero and job preemption is not allowed.

Since the objective is to minimize the makespan, we have to find simultaneously the optimal maintenance locations, the optimal maintenance frequencies and the optimal job schedule, such that the makespan of all jobs are minimized. Using the conventional notation, our problem is denoted as \( R_m/ma/\max C_j \) where \( C_i \) and \( ma \) in the second field represent the completion time of job \( j \) and the maintenance activity respectively.

We define the binary variable \( ( j = 1, 2, \ldots, n \), \( r = 1, 2, \ldots, g \) \( l = 1, 2, \ldots, p \) \( g = 1, 2, \ldots, u \) \( h = 1, 2, \ldots, m \) \) such that \( x_{jrgh} = 1 \) if job \( j \) is scheduled in \( rth \) position after the \( gh \)th maintenance activity on \( lh \)th reset position on machine \( h \) and \( X_{jrgh} = 0 \) otherwise. And also \( y_{gh} \) is a binary variable and \( y_{gh} = 1 \) if \( gh \)th maintenance activity is scheduled on machine \( M_h \) and \( y_{gh} = 0 \) otherwise.

The problem can be formulated as follows:

\[
\text{Min } C_{\max} \tag{1}
\]
\[
\sum_h \sum_{g} \sum_r \sum_l x_{jrgh} = 1 \quad \forall \ j \tag{2}
\]

\[
y_{gh} \geq y_{g+1,h} \quad \forall \ g > 1, \ h \tag{3}
\]
\[
\sum_r \sum_l \sum_j x_{jrgh} \leq y_{gh}.M \tag{4}
\]

\[
C_j \geq (g - 1).t_h \tag{5}
\]

\[
\sum_{j=1}^{f} \sum_{r=1}^{m} \sum_{l=1}^{G} x_{j\cdot r\cdot Tg\cdot h}P_{j\cdot h} + \alpha_{j\cdot h}.(r' - 1) - b_{j\cdot h}.(l' - 1) \tag{6}
\]

\[
\sum_{j=1}^{f} \sum_{r=1}^{m} \sum_{l'=1}^{g'} x_{j\cdot r\cdot Tg'\cdot h}P_{j\cdot h} + \alpha_{j\cdot h}.(r'' - 1) - b_{j\cdot h}.(l'' - 1) \tag{7}
\]

\[
(1 - x_{jrgh})M \quad \forall \ j, r, g, h \tag{8}
\]

\[
C_{\max} \geq C_j \quad \forall \ j \tag{9}
\]

\[
\sum_j \sum_{r} \sum_{g} x_{j\cdot l\cdot g\cdot h} = y_{gh} \quad \forall \ g, h \tag{10}
\]

\[
\sum_j \sum_{r} \sum_{g} x_{j\cdot r\cdot g\cdot h} \leq 1 \quad \forall \ r, g, h \tag{11}
\]

\[
\sum_j \sum_{r} \sum_{g} x_{j\cdot r\cdot l\cdot g\cdot h} \geq (l-1). \tag{12}
\]

\[
x_{jr\cdot lg\cdot h} \geq (1 - x_{jr\cdot lg\cdot h})M \tag{13}
\]

\[
\sum_{j=1}^{f} \sum_{r} \sum_{l} x_{j\cdot r\cdot 1\cdot h} \geq \sum_{j=1}^{f} \sum_{r} \sum_{l} x_{j\cdot r\cdot l\cdot h} \quad \forall \ g, h, r \geq 1 \tag{14}
\]

\[
\sum_{j=1}^{f} \sum_{r} \sum_{g} x_{j\cdot r\cdot l\cdot g\cdot h} \geq \sum_{j=1}^{f} \sum_{r} \sum_{g} x_{j\cdot r\cdot l\cdot 1\cdot h} \quad \forall \ l > 1, h \tag{15}
\]

Constraint (1) is the objective function. Constraint (2) ensures that each job must be assigned to some position \( (r, g, l, h) \). Constraint (3) ensures that the assigned maintenance must be precedes all unassigned maintenance on each machine. Constraint (4) ensure that no job would be done before performing maintenance on machine \( h \). Constraint (5) determine the completion time of each job. Constraint (6) determines the maximum completion time that must be minimized. Constraint (7, 8) assumed that the maintenance activity should be happen in the first position. Constraint (8) ensures that either no job or one job is scheduled in any position \( (r, g, h) \) in which \( r \geq 2 \). Constraint (10, 11 and 12) ensures that on each machine the assigned jobs must be precedes all unassigned jobs. Constraint (13) shows the binary variable decision.
3. Heuristic Algorithms

The problem under study is NP-hard even without consideration of the learning effect, aging effect and multi-maintenance activities. Thus developing efficient heuristic algorithms would be a good approach, although exact algorithms provide optimal solutions, but their long running time make them impractical in most of real-life applications, therefore genetic algorithm and the imperialist competitive algorithm are provided, before developing algorithms, we first utilize a lower bound adjusted form Pinedo for assessing the performance of the heuristic.

Property 1. (Pinedo)[18] let $A=\min_{1\leq k\leq m}^{\pi} p_{j_0}^n a_{j,k}$, then

\[
LB= A \times \max \{ \frac{p_j}{s_1}, \left( \frac{p_1 + p_2}{s_1 + s_2} \right) \sum_{j=1}^{m-1} p_j / \sum_{i=1}^{m-1} s_i, \sum_{j=1}^{n} p_j / \sum_{i=1}^{m} s_i \} 
\]

Is a lower bound of the makespan.

3.1. Genetic Algorithm (GA)

For solving a parallel machine scheduling problem with the effect of aging and learning and multi maintenance activities, a genetic algorithm is proposed. The main procedures of GA for implementation are as outlined below.

3.1.1. Chromosome representation

For a problem of $n$ jobs and $m$ machines, each chromosome has $n$ gens. We randomly generate the $(n,m)$ uniform numbers between $(0,1)$. The ordering of the gens represented the sequencing of jobs. For example a chromosome of $(0.4,0.3,0.9,0.1)$ in a problem with 1 machine would represent as in (fig1):

\[
\begin{array}{cccc}
  j_3 & j_1 & j_2 & j_4 \\
  1 & 0.4 & 1 & 0.3 \\
  1 & 0 & 1 & 0.9 \\
  0.5 & 0.2 & 0.7 & 0.4 \\
\end{array}
\]

Figure 1. job sequencing

The second binary part, decide to assign maintenance activity to gens or not. (Fig 2) illustrated an example of chromosome coding. In this fig four jobs must be performed on 2 machines, each row is assigned to one machine, as it shown $j_2,j_3$ are assigned to machine 1 and the rest to machine 2. Finally we have:

Machine 1: $j_3, j_1$

Machine 2: $j_4$

\[
\begin{array}{cccc}
  j_1 & j_2 & j_3 & j_4 \\
  0 & 1 & 1 & 0 \\
  0.5 & 0.2 & 0.7 & 0.4 \\
  1 & 0.7 & 0.3 & 0.8 \\
  1 & 0.6 & 0.2 & 0.3 \\
\end{array}
\]

Figure 2. chromosome representation

3.1.2. Initialization

Generating a set of initial solution is the first step for implementing GA. The major decision in GA accomplishment is to determine the appropriate population size. If the selected number is too small, we may not be able to achieve a good solution, contrarily the larger number takes too much computational time to achieve a better solution [1]. In this proposed algorithm, the quintuple of first population size is used.

3.1.3. Evaluation

We calculate the fitness value of each chromosome by their objective value.

3.1.4. Crossover

Crossover is the method of merging the genetic information of two individuals. In this study, two methods for crossover operator are used. First is N Point crossover operator which is exchanging of N genes between the chromosomes of two parents. (Fig3 A,B) illustrated the assumed parent, (Fig4) showed five random numbers are generated between $(0,1)$. If generated numbers are less than 0.5, information of first parent will be used otherwise information of second parent would be assigned (Fig 5). The second method that applied is one point crossover operator that generated m uniform numbers between one and n to choose the cutting point (Fig.6 ).
ICA is on the most powerful evolutionary algorithms in the meta heuristic field and the computer simulation based on socio-political evolution that is developed by Atashpaz-Gargari[21]. Imperialist competitive algorithm chart is presented in Fig8. Similar to the other evolution algorithms, the ICA is started with initial random population. Each member of the population is called countries. Some of the best countries based on their cost function are categorized as the imperialist and all other countries form the colonies of the imperialist.

After all empires were formed, the competition between countries begins. First, the colonies in each of the empires start moving toward their relevant imperialist state and change the place in new position. [22]

Each empire that couldn’t be success in competition would be eliminated and is considered as a colony in competition to be assimilated by other imperialist. Remaining of one emperor is the stopping point of the algorithm.

3.1.5. Mutation

The mutation operator randomly selects individual gens and swaps its value. As shown in (Fig 7), two obtained numbers are and so offspring is generated by replacing these two gens.

3.2. Imperialist Competitive Algorithm

![Imperialist competitive algorithm (ICA) flowchart[22]](image)
3.3. Computational Experiments

In this section, we conducted the results of the computational experiments to evaluate the performance of the proposed GA and ICA. We tested only one of the parameters in each level. Each test was repeated five times. The values were considered based on table 1 and 2 for the four parameters used by proposed GA and ICA respectively.

Table 1. Different level of parameters for GA

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Level1</th>
<th>Level2</th>
<th>Level3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration number</td>
<td>100</td>
<td>150</td>
<td>200</td>
</tr>
<tr>
<td>Population size</td>
<td>150</td>
<td>200</td>
<td>250</td>
</tr>
<tr>
<td>Crossover rate</td>
<td>0.9</td>
<td>0.85</td>
<td>0.8</td>
</tr>
<tr>
<td>Mutation rate</td>
<td>0.09</td>
<td>0.12</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Table 2. Different level of parameters for ICA

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Level1</th>
<th>Level2</th>
<th>Level3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of countries</td>
<td>150</td>
<td>200</td>
<td>250</td>
</tr>
<tr>
<td>Number of empire</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>Iteration number</td>
<td>300</td>
<td>400</td>
<td>500</td>
</tr>
<tr>
<td>Revolution possibility</td>
<td>0.1</td>
<td>0.15</td>
<td>0.2</td>
</tr>
</tbody>
</table>

3.4. Taguchy Method

The Taguchi method is a statistical method to achieve robust parameter design [19]. We utilize the Taguchi method to purpose the values of parameters, so table 3 and 4 shows the best value that conducted based on Taguchi method. All the algorithms are coded using MATLAB 7.5 and run on Intel Core 2.50 GHz CPU and 4 GB RAM on windows 7.

Table 3. GA parameters settings

<table>
<thead>
<tr>
<th>GA Parameters</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration number</td>
<td>200</td>
</tr>
<tr>
<td>Population size</td>
<td>150</td>
</tr>
<tr>
<td>Crossover rate</td>
<td>0.9</td>
</tr>
<tr>
<td>Mutation rate</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Table 4. ICA parameters settings

<table>
<thead>
<tr>
<th>ICA Parameters</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of countries</td>
<td>200</td>
</tr>
<tr>
<td>Number of empires</td>
<td>150</td>
</tr>
<tr>
<td>Iteration number</td>
<td>0.9</td>
</tr>
<tr>
<td>Revolution possibility</td>
<td>0.2</td>
</tr>
</tbody>
</table>

4. RPD Analyzing for Proposed Heuristic Algorithms

As it showed in (fig 9) RPDs are zero in small measures and it is an approval of accuracy for three proposed algorithms. Mathematical model is not capable of finding optimal solution in bigger measures. Also it is concluded that there is more gradient in genetic algorithm than ICA algorithm that conducted to ICA could have better efficiency than genetic algorithm.

Figure 9. RPD graph for two proposed algorithm

5. Time Analyzing for Proposed Heuristic Algorithm

Solving time of two heuristic is studied in this section. As illustrated in (fig 10) genetic algorithm solving time has a considerable differences to ICA algorithm which is means ICA could find optimal solution in less time.

Figure 10. Time analyzing for proposed heuristic algorithm
### Table 5. Comparison of results of the model solved by lingo, the proposed GA and ICA

<table>
<thead>
<tr>
<th>no</th>
<th>job</th>
<th>machine</th>
<th>optimal time</th>
<th>mathematical model</th>
<th>Genetic algorithm</th>
<th>ICA algorithm</th>
<th>PSO algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>AVG</td>
<td>Best Time</td>
<td>AVG</td>
<td>Best Time</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>2</td>
<td>00:25:49</td>
<td>32.934</td>
<td>32.934 00:00:5.7</td>
<td>0 32.934</td>
<td>32.934 00:00:0.83</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>3</td>
<td>0:53:12</td>
<td>21.978</td>
<td>21.978 00:00:5.59</td>
<td>0 21.978</td>
<td>21.978 00:00:0.68</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>2</td>
<td>-</td>
<td>58.78</td>
<td>58.78 00:00:11.56</td>
<td>0 58.78</td>
<td>58.78 00:00:1.3</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>3</td>
<td>-</td>
<td>40.868</td>
<td>40.868 00:00:11.65</td>
<td>0 42.068</td>
<td>40.868 00:00:1.36</td>
</tr>
<tr>
<td>5</td>
<td>15</td>
<td>2</td>
<td>-</td>
<td>86.384</td>
<td>86.384 00:00:31.9</td>
<td>0 87.384</td>
<td>87.384 00:00:1.856</td>
</tr>
<tr>
<td>6</td>
<td>15</td>
<td>3</td>
<td>-</td>
<td>56.98</td>
<td>56.78 00:00:32.11</td>
<td>0.017 60.58</td>
<td>58.78 00:00:1.58</td>
</tr>
<tr>
<td>7</td>
<td>30</td>
<td>5</td>
<td>-</td>
<td>72.11</td>
<td>67.97 00:01:42</td>
<td>0.304 78.03</td>
<td>75.97 00:00:4.94</td>
</tr>
<tr>
<td>8</td>
<td>30</td>
<td>7</td>
<td>-</td>
<td>55.397</td>
<td>53.78 00:01:2.159</td>
<td>0.15 59.615</td>
<td>57.78 00:00:3.78</td>
</tr>
<tr>
<td>9</td>
<td>35</td>
<td>5</td>
<td>-</td>
<td>84.187</td>
<td>81.538 00:00:17.7</td>
<td>0.162 91.527</td>
<td>89.538 00:00:4.71</td>
</tr>
<tr>
<td>10</td>
<td>35</td>
<td>7</td>
<td>-</td>
<td>64.478</td>
<td>63.67 00:01:5.43</td>
<td>0.063 67.956</td>
<td>62.78 00:00:4.86</td>
</tr>
<tr>
<td>11</td>
<td>40</td>
<td>5</td>
<td>-</td>
<td>94.708</td>
<td>93.208 00:01:41.2</td>
<td>0.08 103.433</td>
<td>101.208 00:00:5.17</td>
</tr>
<tr>
<td>12</td>
<td>40</td>
<td>7</td>
<td>-</td>
<td>71.952</td>
<td>70.97 00:01:21.37</td>
<td>0.069 78.23</td>
<td>76.27 00:00:5.36</td>
</tr>
<tr>
<td>13</td>
<td>50</td>
<td>7</td>
<td>-</td>
<td>89.125</td>
<td>88.284 00:02:12.47</td>
<td>0.024 95.025</td>
<td>90.538 00:00:7.6</td>
</tr>
<tr>
<td>14</td>
<td>50</td>
<td>12</td>
<td>-</td>
<td>54.457</td>
<td>53.78 00:01:28.19</td>
<td>0.341 63.54</td>
<td>59.78 00:00:7.264</td>
</tr>
<tr>
<td>15</td>
<td>60</td>
<td>7</td>
<td>-</td>
<td>105.913</td>
<td>104.584 00:02:5.32</td>
<td>0.063 114.008</td>
<td>110.208 00:00:8.18</td>
</tr>
<tr>
<td>16</td>
<td>60</td>
<td>12</td>
<td>-</td>
<td>68.316</td>
<td>65.67 00:01:19.45</td>
<td>0.201 72.77</td>
<td>69.67 00:00:8.63</td>
</tr>
<tr>
<td>17</td>
<td>70</td>
<td>12</td>
<td>-</td>
<td>78.784</td>
<td>74.67 00:01:40.86</td>
<td>0.275 82.897</td>
<td>79.67 00:00:7.89</td>
</tr>
<tr>
<td>18</td>
<td>70</td>
<td>15</td>
<td>-</td>
<td>65.932</td>
<td>62.58 00:01:35.89</td>
<td>0.28 71.092</td>
<td>66.98 00:00:10.03</td>
</tr>
<tr>
<td>19</td>
<td>100</td>
<td>12</td>
<td>-</td>
<td>67.33</td>
<td>64.08 00:01:38.14</td>
<td>0.254 74.314</td>
<td>70.97 00:00:7.21</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>15</td>
<td>-</td>
<td>90.98</td>
<td>89.038 00:02:24.42</td>
<td>0.109 96.241</td>
<td>94.384 00:00:13.87</td>
</tr>
</tbody>
</table>
6. Conclusions

In this paper we investigated unrelated parallel machine scheduling problems simultaneously with the effects of learning and aging and multi-maintenance activities. We aimed to find the optimal maintenance locations, the optimal maintenance frequencies and the optimal job schedule to minimize the makespan. We propose polynomial time algorithm to solve the problem. Three heuristic algorithms were proposed and the result proved the better efficiency of ICA algorithm. Further research may investigate the problem with deteriorating multi-maintenance activities and optimizing other performance measures.

REFERENCES