Enhancement of Multiprocessors Job Shop Scheduling with Heterogeneous Workers Using Particle Swarm Optimization

Shun-chi YU*

China-Asean International College, Dhurakij Pundit University, Thailand 110/1-4 Prachachuen Rd., Laksi, Bangkok 10210
*Corresponding author

Keywords: Heterogeneous workers, PSO, Multiprocessors job shop, Algorithm.

Abstract. To present employees’ job processing time with fuzzy numbers, this study utilizes a stepwise function applied to forming maximum completion time as an independent variable to simulate the real work situations. Given that multiprocessor job shop scheduling is applicable to the actual production scenarios, this study chooses multiprocessor job shop with heterogeneous workers scheduling as the subject. This study oriented to the multiprocessor job shop scheduling takes the heterogeneous workers with related subsidies as the constraining factors and endeavors to seek the best solutions to the problem. Finally, simulated data confirm the effectiveness and robustness of the proposed algorithm. The data test results indicate that PSO is a significant heuristic algorithm for the attempted problems.

Introduction

In recent years, the aged population over the age of 65 in the production line attracts more attention, as companies have been confronted with the deficiencies of skilled labors. Nevertheless, well-developed medical care and health conditions have prolonged the contribution of the aged workers. In 2002, the European Union proposed the concept of “active ageing”, the keynote of which is to extend the working life for the middle-aged and older workers (WHO, 2002). The classification of production management combines methods of distributing operational activities to reduce makespan cost, inventory cost, wages and other operational costs. In order to minimize those cost, the employment of the older and disabled becomes a flexible productive strategy. Meanwhile, many useful approaches exist for solving scheduling problems, and have varied results.

Literature Review

As far as all the scheduling problems are concerned, multiprocessor job shop scheduling is seen as the most common production scenario in the real manufacturing industry, such as in the mould-making, steel mill, and accessories shop processing environments. Thus, a multi-objective problem considering uncertainty and flexibility of job sequence in an automated flexible job shop (AFJS) is considered using manufacturing simulation, results show the optimal resource assignment and optimal job sequence with efficiency and productivity minimizes makespan, the number of late jobs, total flow time and total weighted flow time in an automated flexible job shop (Amiri, Shirazi & Tajdin, 2019; Wua, Shena, & Li, 2019).

As for heterogeneous workers, most studies considered unrelated machines for solving various scheduling problems. Scheduling considering learning effect is applied to solve a comparison of two stage-based hybrid algorithms for a batch scheduling problem in hybrid flow shop (Sakawa & Kubota, 2001). To develop a practical solution applicable in real world conditions, this investigation considers the essential job shop scheduling problem given heterogeneous workers, including the old-aged and disabled with related subsidies for constraints. The subsidies, obtained when companies complied with government’s labor policy for caring the week, can complement the deficiency of the old-aged and disabled compared with that of regular workers.
Regarding uncertainty processing time, Xie and Chen (2018) posited that job processing time was not always accurately as a time quota in a production system, particularly in the complex products manufacturing process. In practice, processing times can be more accurately represented as intervals with the most probable completion time somewhere near the middle of the interval. Thus, this study utilizes fuzzy processing time for approximating the real world, instead of crisp value.

Methodology
Kennedy and Eberhart (1997) developed a binary particle swarm optimization (BPSO) algorithm by constructing a discrete space to improve PSO’s former research limitation in continuous space. Binary particle swarm optimization using binary notation can present various discretized solutions and solve 0-1 style problems. Particle swarm optimization can increase the feasibility of solving the flow shop scheduling problems in every field. The parameter settings in the proposed algorithm are decided during pre-test. All tests are performed on a personal computer with Intel® Core™ i7-6700K CPU @ 4.00GHz, 16.0 GB RAM. Finally, the test results are compared and analyzed. Parameters related to PSO algorithms include: where velocity \((v_{max})\) is 6, inertia \((I)\) is 1, weights of \(c_1, c_2, c_3\) are 2, 2, 1, number of loop \((nL)\) is 100, number of particles \((nP)\) the number of particles in each iteration is 20, and iterations are 100 (loop).

Equations
\[
\begin{align*}
n &= \text{Total number of jobs} \\
m &= \text{Total number of machines} \\
o &= \text{Total number of operations to finish} \\
J_i &= \text{Job of number } i, i=1, 2, ..., n \\
O_{ij} &= \text{The } j\text{-th operation of } J_i \\
M_k &= \text{Machine of number } k \\
O_{ijk} &= \text{ } O_j \text{ is processed on } M_k \\
FFT_k &= \text{Number of free float time on } M_k \\
IFT_k &= \text{Number of interfering float time on } M_k \\
no_k &= \text{ } \text{Assigned number of operations on } M_k \\
uc_0 &= \text{ } \text{Unit cost generated by incremental makespan} \\
su_d &= \text{ } \text{The amount of subsidy per hour for hiring a disabled worker} \\
su_o &= \text{ } \text{The amount of subsidy per hour for hiring an old-aged worker} \\
e_r &= \text{Number of hired regular worker} \\
e_d &= \text{Number of hired disabled worker} \\
e_o &= \text{Number of hired old-aged worker} \\
C_{um}(e_r, e_d, e_o) &= \text{Makespan considering number of hired regular, old-aged and disabled worker} \\
Q_i &= \text{Number of operations to finish } J_i \\
L &= \text{A big number} \\
S_{ijk} &= \text{Starting time of } O_{ijk} \\
p_{ijk} &= \text{Processing time of } O_{ijk} \\
F_{ijk} &= \text{Completion time of } O_{ijk} \\
p_{ijkw} &= \text{Processing time for regular worker to process } O_{ijk} \\
p_{ijkdw} &= \text{Processing time for disabled worker to process } O_{ijk}
\end{align*}
\]
\( P_{ijkw} \) = Processing time for old-aged worker to process \( O_{ijk} \)

\[
X_{ijk} = \begin{cases}
1, & \text{if } O_{ij} \text{ is processed on } M_k \\
0, & \text{otherwise}
\end{cases}
\]

\[
Y_{ijklk} = \begin{cases}
1, & \text{if } O_{ij} \text{ is processed before } O_{ijkl} \text{ on } M_k \\
0, & \text{otherwise}
\end{cases}
\]

\[
Z_{ijkw} = \begin{cases}
1, & \text{if } M_k \text{ is processed by regular worker} \\
0, & \text{otherwise}
\end{cases}
\]

\[
Z_{ijkw} = \begin{cases}
1, & \text{if } M_k \text{ is processed by old-aged worker} \\
0, & \text{otherwise}
\end{cases}
\]

\[
Z_{ijkw} = \begin{cases}
1, & \text{if } M_k \text{ is processed by disabled worker} \\
0, & \text{otherwise}
\end{cases}
\]

**Mathematical Model**

**Objective formulation:**

\[
\min Z = C_{\max}(e_r, e_d, e_o) \cdot (uc_0 \cdot m - su_d \cdot e_d - su_o \cdot e_o) \quad (1)
\]

Subject to:

\[
C_{\max}(e_r, e_d, e_o) = \max(F_{ijk}) \quad \forall i = 1, 2, \ldots, n; j = 1, 2, \ldots, Q_i; k = 1, 2, \ldots, m \quad (2)
\]

\[
F_{ijk} = S_{ijk} + p_{ijk} \quad \forall i = 1, 2, \ldots, n; j = 1, 2, \ldots, Q_i; k = 1, 2, \ldots, m \quad (3)
\]

\[
F_{i(j-l)k} \leq S_{ijk} \quad \forall i = 1, 2, \ldots, n; j = 1, 2, \ldots, Q_i; k = 1, 2, \ldots, m \quad (4)
\]

\[
p_{ijk} = \begin{cases}
p_{ijkw}, & \text{if } Z_{ijkw} = 1 \\
p_{ijkw}, & \text{if } Z_{ijkw} = 1 \\
p_{ijkdw}, & \text{if } Z_{ijkdw} = 1, \forall i = 1, 2, \ldots, n; j = 1, 2, \ldots, Q_i; k = 1, 2, \ldots, m
\end{cases} \quad (5)
\]

\[
Z_{ijkw} + Z_{ijkw} + Z_{ijkdw} = 1 \quad \forall i = 1, 2, \ldots, n; j = 1, 2, \ldots, Q_i; k = 1, 2, \ldots, m 
\]

\[
F_{ijk} - L(1-Y_{ijklk}) \leq S_{ijk}, \forall i = 1, 2, \ldots, n; j = 1, 2, \ldots, Q_i; k = 1, 2, \ldots, m; l = 1, 2, \ldots, n; q = 1, 2, \ldots, Q_l; i \neq l \quad (7)
\]

\[
F_{ijk}(1-Y_{ijklk}) - L \cdot Y_{ijklk} \leq S_{ijk}, \forall i = 1, 2, \ldots, n; j = 1, 2, \ldots, Q_i; k = 1, 2, \ldots, m; l = 1, 2, \ldots, n; q = 1, 2, \ldots, Q_o; i \neq l \quad (8)
\]

\[
\sum_k X_{ijk} = 1, \forall i = 1, 2, \ldots, n; j = 1, 2, \ldots, Q_i; k = 1, 2, \ldots, m \quad (9)
\]

The meaning of all objectives and constraints are explained as follows:

Equ.(1): this model minimizes the product of makespan considering number of hired regular, old-aged and disabled worker and deduction of subsidies of the disabled and old-aged worker from unit cost generated by incremental makespan.
Equ.(2): makespan considering number of hired regular, old-aged and disabled worker represents the maximum completion time among all the machines.

Equ.(3): completion time of $O_{ij}$ is the sum of its starting time and processing time.

Equ.(4): as for job $O_{ij}$, the starting time of the $j$-th operation exceeds the completion time of the $(j-1)$-th operation.

Equ.(5): the processing time of $O_{ij}$ is decided by different workers on machine $M_k$.

Equ.(6): $O_{ij}$ is processed only by regular, disabled, and old-aged worker on machine $M_k$.

Equ.(7): $O_{ij}$ is processed before $O_{lj}$, and thus the completion time of $O_{ij}$ is smaller than the starting time of $O_{lj}$.

Equ.(8): $O_{ij}$ is processed after $O_{lj}$, and thus the completion time of $O_{lj}$ is smaller than the starting time of $O_{ij}$.

Equ.(9): $O_{ij}$ is only processed on one machine once.

Results

According to the feature of the attempted problem, this study adopts various scale number of machines and jobs. The range from small to large scale number is $m=3, n=3$; $m=7, n=7$; $m=11, n=11$; $m=17, n=17$; $m=20, n=20$. Due to limitation of content, this study lists $m=3, n=3$ for small scale, and $m=20, n=20$ for large scale demonstration, respectively. The item “E” of Table 2 represents the number of hired heterogeneous workers, and “R”, “O”, “D” in dispatching rule means the assigned regular, old-aged, disabled worker respectively, and the item “ms” means microsecond of CPU.

As shown in Table 1, the assigned sequence considering degree of slack is $M_2, M_1, M_3$. When hiring a heterogeneous worker, the best assignment is RDR; when hiring two, the best assignment is DDR; when hiring three, the best assignment is DDD, the results demonstrate that the hiring of heterogeneous workers surely decreases costs with efficiency. Since the lowest cost is $1791.75$, the cost increases with the hired number of heterogeneous workers. Therefore, under the condition of $m=3, n=3$, it is recommended to hire an unrelated worker to operate in machine 2, and regular workers to operate the rest machines, which generates the most efficiency of saving costs.

Table 2. The improvement ratios of optimal solution and makespan.

<table>
<thead>
<tr>
<th>(m,n)</th>
<th>$E$</th>
<th>Dispatching rule</th>
<th>Optimum</th>
<th>Improvement ratios(%)</th>
<th>Cmax</th>
<th>Improvement ratios(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(3,3)</td>
<td>0</td>
<td>RR</td>
<td>2151.5</td>
<td>0.00%</td>
<td>51.25</td>
<td>0.00%</td>
</tr>
<tr>
<td>(3,3)</td>
<td>1</td>
<td>RR</td>
<td>2151.5</td>
<td>0.00%</td>
<td>51.25</td>
<td>0.00%</td>
</tr>
<tr>
<td>(7,7)</td>
<td>0</td>
<td>RR</td>
<td>117679</td>
<td>0.00%</td>
<td>117.5</td>
<td>0.00%</td>
</tr>
<tr>
<td>(7,7)</td>
<td>1</td>
<td>RR</td>
<td>117679</td>
<td>0.00%</td>
<td>117.5</td>
<td>0.00%</td>
</tr>
<tr>
<td>(11,11)</td>
<td>0</td>
<td>RR</td>
<td>38184</td>
<td>0.00%</td>
<td>248</td>
<td>0.00%</td>
</tr>
<tr>
<td>(11,11)</td>
<td>2</td>
<td>RR</td>
<td>38184</td>
<td>0.00%</td>
<td>248</td>
<td>0.00%</td>
</tr>
<tr>
<td>(17,17)</td>
<td>0</td>
<td>RR</td>
<td>97236</td>
<td>0.00%</td>
<td>408.5</td>
<td>0.00%</td>
</tr>
<tr>
<td>(17,17)</td>
<td>3</td>
<td>RR</td>
<td>97236</td>
<td>0.00%</td>
<td>408.5</td>
<td>0.00%</td>
</tr>
<tr>
<td>(20,20)</td>
<td>0</td>
<td>RR</td>
<td>1456163</td>
<td>0.00%</td>
<td>522</td>
<td>0.00%</td>
</tr>
<tr>
<td>(20,20)</td>
<td>1</td>
<td>RR</td>
<td>1456163</td>
<td>0.00%</td>
<td>522</td>
<td>0.00%</td>
</tr>
</tbody>
</table>
Due to the limit of content, this study had tested all conditions from small to large scale number. Under the condition of m=20, n=20, it is recommended to hire a disabled worker to operate in machine 1, and regular workers to operate the rest machines, which generates the most efficiency of saving costs.

As shown in Table 2, the improvement ratios of optimum are up to 16.67%, which means the percentage of cost-down under the proposed dispatching rule, with tolerable makespans compared with all regular workers hired. Test results indicate that improvement ratios are definite and outstanding in each test set.

**Conclusion**

This study significantly outperforms and develops PSO to solve the multiprocessors job shop scheduling problem, then compares various scale of solutions for saving costs. According to the test results and analysis, this study infers:

1. The effectiveness improvement ratio of hiring heterogeneous workers compared with regular ones in m=3, n=3 is 16.7%, and 5.5% in m=7, n=7, 7.9% in m=11, n=11, 0.5% in m=17, n=17, 1.6% in m=20, n=20. This verifies that the proposed approach significantly outperforms in terms of effectiveness and costs.

2. The results demonstrate that the effectiveness and efficiency of the proposed method clearly exceed than that of manual production scheduling.

3. Under the various scales set in this study, PSO has outstanding solving efficiency.

**Acknowledgement**

This research was financially supported by the National Science Foundation.

**References**


