Abstract. This paper introduces a new fuzzy greedy heuristic (FGH) for the job-shop scheduling problem (JSP) with the mean total completion time (MTCT) criterion. The heuristic consists of two phases: building a candidate list of operations, and constructing a schedule. The candidate list is formed from all possible unscheduled operations, each of which can be incorporated into the partial schedule under construction without causing infeasibility. Computational experiments using a wide range of standard benchmark problems indicate that the proposed method is fast and very effective. In: Sheibani K (ed). Proceedings of the 1st International Conference on Applied Operational Research – ICAOR (2008), pp 72–76. Lecture Notes in Management Science Vol. 1. ISSN 2008-0050.

Keywords: Combinatorial optimisation, heuristics, job-shop scheduling.

1 Introduction

The job shop problem (JSP) is one of the most challenging classical scheduling problems and reflects real operation of several industries. In general form the JSP can be stated as follows. There are $n$ jobs to be processed by a finite set of $m$ machines on which each job visits each machine at most once in its own predetermined route. There are numerous distinct scheduling goals (Mellor, 1966). One of the usual objectives is to minimise the mean total completion time of the jobs. Very simple special cases of the JSP are already strongly NP-hard (Garey et al, 1976). Exact methods have an exponential time bound and have successfully solved only very small size instances. Hence, approximate methods are generally considered to be the only practical way to solve most real-life problems. Many heuristics have been proposed for the problem. Most of them are based on priority dispatching rules (Panwalker and Iskander, 1977, Haupt, 1989). Other approaches have also been experimented for the JSP. The collections of survey papers (Pinson, 1995,
Holthaus and Rajendran, 1997, Jain and Meeran, 1999) and books (French, 1982, Pinedo, 2005) summarise various developments in the subject.

This paper introduces a new fuzzy greedy heuristic (FGH) for the JSP with the mean total completion time (MTCT) criterion. Computational experiments using a wide range of standard benchmark problems indicate that the proposed method is fast and very effective. The concluding remarks contain some suggestions for further research.

2 Methodology

The proposed heuristic consists of two phases: building a candidate list of operations in priority order and then constructing a schedule. The candidate list is formed from all possible unscheduled operations, each of which can be incorporated into the partial schedule under construction without causing infeasibility. The priority of the operations in the list is determined according to an evaluation function as shown in Equation (1). This is a modification of the general formulae of the families of fuzzy membership functions described in (Klir and Yuan, 1995). We refer to this as a fuzzy greedy evaluation function. The selected operation from the updated candidate list is inserted at the earliest available feasible time slot in the schedule under construction.

\[
\mu(x) = \frac{1}{1 + \lambda^2 \left( \frac{1 - \lambda}{\lambda} x - \theta \right)^2} 
\]

In this equation, \( x \) is a generic variable. The parameter \( \theta \) is a basic measure for evaluating the priority to be assigned to \( x \). The parameter \( \lambda \) is a tuning parameter that is chosen by experimentation such that \( 0 \leq \lambda < 1 \) to adjust \( \theta \).

Let \( P_{ij} \) be the processing time of job \( j \) on machine \( i \) corresponding to operation \( o(i,j) \). Define \( T_j = \sum_{i=1}^{\mathcal{M}} P_{ij} \) the total processing time of job \( j \) on each machine, \( \bar{P} = (1/\mathcal{M}) \sum_{j=1}^{\mathcal{N}} T_j \) the average of the processing times, and \( \bar{T} = (1/\mathcal{N}) \sum_{j=1}^{\mathcal{N}} T_j \) the average of the total processing times. Here we define \( x \) and \( \theta \) following two distinct strategies S1 and S2. First, we represent \( x \), with \( x_{i,j} \) corresponding to operation \( o(i,j) \).

<table>
<thead>
<tr>
<th>Table 1. Strategies S1 and S2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy</td>
</tr>
<tr>
<td>(S1)</td>
</tr>
<tr>
<td>(S2)</td>
</tr>
</tbody>
</table>

The proposed evaluation function has the following properties: \( \mu(\lambda \theta/(1-\lambda)) = 1 \) and \( 0 < \mu(x_{i,j}) < 1 \) for all \( x_{i,j} = \lambda \theta/(1-\lambda) \). This implies that scheduling the job with \( x_{i,j} \) closest numerically to \( \lambda \theta/(1-\lambda) \) should be given higher priority. The steps of the heuristic are as follows:
The FGH Heuristic

1) Calculate $\mu(x_{i,j})$ for each operation;
2) Arrange the candidate list of operations;
3) Select the next operation with the highest priority from the candidate list and insert it at the earliest available feasible time slot on the schedule under construction;
4) Repeat steps 2 and 3 until all operations are scheduled.

The computational complexity of the proposed heuristic is the same as that of the corresponding greedy algorithm Sheibani (2005). It should be noted that there is some additional computational effort independent of $m$ and $n$ for tuning the parameter $\lambda$ to obtain the best performance of the heuristic. For more details on methodology we refer the interested reader to Sheibani (2008).

3 Experimental Results

The proposed heuristic was implemented in C++ code. The test problems are part of an extensive set of standard problems taken from Taillard’s (1993) benchmark instances (http://mistic.heig-vd.ch/taillard/problemes.dir/problemes.html accessed 25 April 2008). Two adaptations of the heuristic are presented as FGH(S1) and FGH(S2) corresponding to strategies S1 and S2.

We introduced the tuning parameter $\lambda$ to achieve a good performance of the proposed heuristic. The efficiency of the FGH heuristic depends greatly on the choice of the parameter $\lambda$ in an effective range. If the range is too small, the probability that it includes the best $\lambda$ value will be low. If it is too large, the algorithm may waste computational resources and so the waiting time for an improvement could be too long. Figure 1 illustrates the effect of different values of $\lambda$ on the computational performance of the proposed heuristic using strategy S2.

![Fig. 1. The FGH(S2) for a range of different values of $\lambda$ for problem ta55 (50×15)](image-url)
The performance of the proposed heuristic in comparison with two classical dispatching rules SPT (shortest processing time first), which has been commonly regarded as one of the most effective heuristics for this problem (Holthaus and Rajendran, 1997) and LPT (longest processing time first) is reported in Table 1.

Table 2. A comparison of the FGH heuristic with other heuristics.

<table>
<thead>
<tr>
<th>Problem</th>
<th>FGH(S1)</th>
<th>FGH(S2)</th>
<th>SPT</th>
<th>LPT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Avg</td>
<td>Stdev</td>
<td>Avg</td>
<td>Stdev</td>
</tr>
<tr>
<td>15×15</td>
<td>1295.20</td>
<td>26.11</td>
<td>1240.91</td>
<td>38.93</td>
</tr>
<tr>
<td>30×20</td>
<td>2142.56</td>
<td>47.23</td>
<td>2026.80</td>
<td>45.87</td>
</tr>
<tr>
<td>50×15</td>
<td>2588.95</td>
<td>60.99</td>
<td>2311.01</td>
<td>54.56</td>
</tr>
</tbody>
</table>

The results show that FGH(S2) is superior to all of the methods for all of the problems of varying sizes in term of the average solution obtained. On average, FGH(S2) has an error of 7.37%, 11.01%, and 12.57% less than FGH(S1), SPT, and LPT, respectively.

We investigated the performance of the proposed heuristic when the fuzzy part is turned off. In this form, we reduce both adaptations FGH(S1) and FGH(S2) to corresponding greedy versions by taking either minimum or maximum values of $x_{ij}$. We refer these as S(S) (shortest strategy first) and L(S) (longest strategy first). The results are summarised in Table 2.

Table 3. The performance of the FGH heuristic when the fuzzy part is turned off.

<table>
<thead>
<tr>
<th>Problem</th>
<th>S(S1)</th>
<th>S(S2)</th>
<th>L(S1)</th>
<th>L(S2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Avg</td>
<td>Stdev</td>
<td>Avg</td>
<td>Stdev</td>
</tr>
<tr>
<td>15×15</td>
<td>1371.27</td>
<td>48.37</td>
<td>1307.84</td>
<td>39.81</td>
</tr>
<tr>
<td>30×20</td>
<td>2248.18</td>
<td>40.98</td>
<td>2098.98</td>
<td>57.40</td>
</tr>
<tr>
<td>50×15</td>
<td>2698.13</td>
<td>48.21</td>
<td>2380.48</td>
<td>68.21</td>
</tr>
</tbody>
</table>

The results show that the fuzzy adaptations significantly improve relative performance over greedy versions for all of the problems of varying sizes in term of the solution obtained. In a sense, the fuzzy adaptations can be said to dominate the greedy versions. On average, FGH(S1) and FGH(S2) have an error of 4.12% and 3.98% respectively less than the best performing corresponding greedy versions L(S1) and S(S2). However, the results for S(S2) are still superior to those of the SPT rule.
4 Concluding Remarks

This paper has provided an overview of a new heuristic for the job shop scheduling problem (JSP) with minimisation of the mean total completion time as the objective. We examined the effectiveness and the efficiency of the proposed heuristic on a wide range of benchmark problems of varying sizes. The developed fuzzy greedy heuristic (FGH) is superior to the SPT rule, which has been commonly regarded as one of the most effective heuristics for this problem. For future research we believe that the following topics are potentially useful: 1) developing efficient adaptations of the proposed heuristic; 2) extending our method to other objectives; 3) developing efficient methods using the fuzzy greedy evaluation concept in other areas of combinatorial optimisation.

References