Lecture 9

Scheduling and Vehicle Routing

Marco Chiarandini
1. Scheduling
   Group Shop Scheduling

2. Appendix
Outline

1. Scheduling
2. Appendix
Group Shop Problems (GSP)

- Multi-stage, Multi-machine
- A partial order is given for the operations of each job.
- Special cases:
  - Open Shop: no restriction on the order of jobs through the machines
  - Job Shop: the order of jobs through the machines is fixed and

- \( m \) machines \( \mathcal{M} = \{M_1, \ldots, M_m\} \)
- \( n \) jobs \( \mathcal{J} = \{J_1, \ldots, J_n\} \)
- \( m \) operations per job \( J_i = \{o_{i1}, \ldots, o_{im}\} \)
- Operation \( o_{ij} \) of \( J_i \) has to be processed by \( M_j \)
- Precedence constraints: collection of sets \( G_i = \{g_{i1}, \ldots, g_{il(i)}\} \) with total order \( g_{i1} \prec g_{i2} \prec \cdots \prec g_{il(i)} \)
- Processing time \( p_{ij} \) for each operation \( o_{ij} \)
- Ignored features: setup times, due dates, release dates, pre-emption, operations on more than one machine
- Minimize makespan \( C_{max} \)
Example (GSP)

- 3 machines, 3 jobs
- \( J_1 = \{o_{11}, o_{12}, o_{13}\} \), \( J_2 = \{o_{21}, o_{22}, o_{23}\} \), \( J_3 = \{o_{31}, o_{32}\} \)
- \( G_1 : \{o_{11}, o_{12}\} \prec \{o_{13}\} \) // \( G_2 : \{o_{21}\} \prec \{o_{22}, o_{23}\} \) // \( G_3 : \{o_{31}, o_{32}\} \)

<table>
<thead>
<tr>
<th>( p_{ij} )</th>
<th>( i = 1 )</th>
<th>( i = 2 )</th>
<th>( i = 3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( j = 1 )</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>( j = 2 )</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>( j = 3 )</td>
<td>1</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>

Gantt Chart of a feasible schedule \((o_{21}, o_{31}, o_{11}; o_{12}, o_{22}, o_{32}; o_{13}, o_{23})\)
Local Search Algorithms

Construction heuristics: Candidate list dispatching rules, NEH

Solution representation: $m$ consecutive strings each being a permutation of $m_k = |M_k|$ operations on a specific machine:

$$\pi = (\pi_1, \ldots, \pi_m) \quad \pi_k = (\pi_k(1), \ldots, \pi_k(m_k))$$

Operations can be represented as $O = \{1, \ldots, o\}$. Each job consists of a sequence of operations indexed consecutively.

In the previous example: $\pi = (\pi_1, \pi_2, \pi_3)$ where $\pi_1 = (5, 7, 1)$, $\pi_2 = (2, 4, 8)$ $\pi_3 = (3, 5)$

Neighborhood structures: Insertion and Swap of critical operations

Metaheuristics:
- Beam-ACO for Open Shop
- Novicki and Smutnicki Tabu Search for Job Shop
Concluding Remarks

Issues on Real World Scheduling

▶ Presence of release dates, setups, pre-emptions, dynamic process, stochastic data, multi-objectives
▶ For a scheduling system being successful, the scheduling algorithms must solve the particular real problem not a simplified abstraction.
▶ The scheduling system must be fast and it includes user interfaces and data sources.
▶ Rescheduling is a very important task is real scenarios, therefore algorithms need to support dynamic scheduling.

Other Manufacturing Scheduling Problems

▶ Project planning
▶ Parallel machine systems
▶ Flexible assembly systems, Lot sizing
▶ Supply chain
1. Scheduling

2. Appendix
## Construction heuristics for Single Machine Scheduling

### Dispatching rules

<table>
<thead>
<tr>
<th>RULE</th>
<th>DATA</th>
<th>OBJECTIVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earliest Release Date</td>
<td>$r_j$</td>
<td>Variance in throughput times</td>
</tr>
<tr>
<td>Earliest Due Date</td>
<td>$d_j$</td>
<td>Maximum Lateness</td>
</tr>
<tr>
<td>Minimum Slack</td>
<td>$d_j$</td>
<td>Maximum Lateness</td>
</tr>
<tr>
<td>Longest Processing Time</td>
<td>$p_j$</td>
<td>Load balancing over parallel machines</td>
</tr>
<tr>
<td>Shortest Processing Time</td>
<td>$p_j$</td>
<td>Sum of completion times</td>
</tr>
<tr>
<td>Weighted Shortest Processing Time</td>
<td>$p_j, w_j$</td>
<td>Weighted sum of completion times</td>
</tr>
<tr>
<td>Critical Path</td>
<td>$p_j, prec$</td>
<td>Makespan</td>
</tr>
<tr>
<td>Largest Number of Successors</td>
<td>$p_j, prec$</td>
<td>Makespan</td>
</tr>
<tr>
<td>Service in Random Order</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Shortest Setup Times First</td>
<td>$s_{jk}$</td>
<td>Makespan and throughput</td>
</tr>
<tr>
<td>Least Flexible Job First</td>
<td>$M_j$</td>
<td>Makespan and throughput</td>
</tr>
<tr>
<td>Shortest Queue at the Next Operation</td>
<td>–</td>
<td>Machine Idleness</td>
</tr>
<tr>
<td>Adjusted Urgency</td>
<td>$d_j, p_j$</td>
<td>Sum weighted tardiness</td>
</tr>
</tbody>
</table>
Critical Path

Critical path formed by critical operations, i.e., those whose starting time cannot be postponed without changing the value of the objective function.

Earliest starting time

Forward Procedure:

\[ es_{\pi(1)j} = \sum_{h=1}^{j-1} p_{\pi(1)h} \]
\[ es_{\pi(i)1} = \sum_{h=1}^{i} p_{\pi(i-1)1} \]

Compute inductively

\[
\begin{align*}
es_{\pi(i)j} &= \max\{C_{\pi(i-1)j}, C_{\pi(i)(j-1)}\} \\
C_{\pi(i)j} &= es_{\pi(i)j} + p_{\pi(i)j} \\
C_{\text{max}} &= \max\{C_{\pi(i)j}\}
\end{align*}
\]

Latest starting time

Backward Procedure:

\[ lc_{\pi(n)m} = C_{\text{max}} \]
\[ ls_{\pi(n)m} = lc_{\pi(n)m} - p_{\pi(n)m} \]

Compute Inductively

\[
\begin{align*}
lc_{\pi(i)j} &= \min\{ls_{\pi(i),(j+1)}, ls_{\pi(i+1),j}\} \\
ls_{\pi(i)j} &= lc_{\pi(i)j} - p_{\pi(i)j}
\end{align*}
\]

Critical operation iff \( es_{ij} = ls_{ij} \).

Two consecutive critical operations either belong to the same machine or to the same job.
A Starting Scheme for the Course Project

- Statement of the problem
  - Short literature review
  - Formalization: constraints and objectives
- Generation of test instances
- Implementation of a solution checker
- Construction heuristics
- Iterative Improvements
- Metaheuristics
- Evaluation by tests
Construction of Test Instances

- Real world data set
  - match the characteristics of interest
  - often are of difficult access; limited in number; missing information
- Random variants of real data
  - problem characteristics under control; unlimited in number;
- Published and on-line libraries
  - often not related to real world
  - not representative of the characteristics of interest
- Randomly generated instances
  - quick and often valid; problem characteristics under control; unlimited in number; sometime it is possible to know the optimal solution
  - may be misleading; correlation vs independence of parameters