LOCAL SEARCH METHODS
APPLICATIONS AND ENGINEERING

Lecture 9

Scheduling and Vehicle Routing

Marco Chiarandini

Outline

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

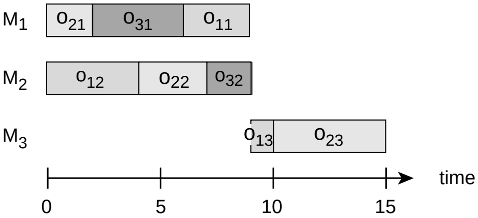
- m machines $\mathcal{M} = \{M_1, \ldots, M_m\}$
- n jobs $\mathcal{J} = \{J_1, \ldots, J_n\}$
- m operations per job $J_i = \{o_{i1}, \ldots, o_{im}\}$
- Operation o_{ij} of J_i has to be processed by M_j
- Precedence constraints: collection of sets $G_i = \{g_{i,1}, \cdots, g_{i,l(i)}\}$ with total order $g_{i,1} \prec g_{i,2} \prec \ldots \prec g_{i,l(i)}$
- 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
- $J_1 = \{o_{11}, o_{12}, o_{13}\}$, $J_2 = \{o_{21}, o_{22}, o_{23}\}$, $J_3 = \{o_{31}, o_{32}\}$
- $G_1 : \{o_{11}, o_{12}\} \prec \{o_{13}\}$ // $G_2 : \{o_{21}\} \prec \{o_{22}, o_{23}\}$ // $G_3 : \{o_{31}, o_{32}\}$

p_{ij}	$i = 1$	$i = 2$	$i = 3$
$j = 1$	3	2	4
$j = 2$	4	3	2
$j = 3$	1	5	0

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:

$$\pi = (\pi_1, \dots \pi_m) \qquad \pi_k = (\pi_k(1), \dots, \pi_k(m_k))$$

Operations can be represented as $O = \{1, \dots, o\}$. 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

Outline

1. Scheduling

2. Appendix

Construction heuristics for Single Machine Scheduling

Dispatching rules

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

Critical Path

Critical path formed by critical operations, ie, 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

$es_{\pi(i)j} = \max\{C_{\pi(i-1)j}, C_{\pi(i)(j-1)}\}$

$C_{\pi(i)j} = es_{\pi(i)j} + p_{\pi(i)j}$

$C_{max} = \max\{C_{\pi(i)j}\}$

Latest starting time

Backward Procedure:

$lc_{\pi(n)m} = C_{max}$

$ls_{\pi(n)m} = lc_{\pi(n)m} - p_{\pi(n)m}$

Compute Inductively

$lc_{\pi(i)j} = \min\{ls_{\pi(i),(j+1)}, ls_{\pi(i+1),j}\}$

$ls_{\pi(i)j} = lc_{\pi(i)j} - p_{\pi(i)j}$

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