DM87
SCHEDULING,
TIMETABLING AND ROUTING

Lecture 13

Resource Constrained Project Scheduling. Reservations and Timetabling

Marco Chiarandini

Outline

Outline

Preprocessing: Temporal Analysis

- Precedence network must be acyclic
- Heads r_j and Tails q_j $\Leftarrow$ Longest paths $\Leftarrow$ Topological ordering (deadlines d_j can be obtained as $UB - q_j$)

Preprocessing: constraint propagation

1. conjunctions $i \longrightarrow j$ $\quad S_i + p_i \leq S_j$
   [precedence constrains]
2. parallelity constraints $i \,||\, j$ $\quad S_i + p_i \geq S_j$ and $S_j + p_j \geq S_i$
   [time windows $[r_j, d_j]$,$[r_l, d_l]$ and $p_l + p_j > \max\{d_l, d_j\} - \min\{r_l, r_j\}$]
3. disjunctions $i - j$ $\quad S_i + p_i \leq S_j$ or $S_j + p_j \leq S_i$
   [resource constraints: $r_{jk} + r_{lk} > R_k$]

N. Strengthenings: symmetric triples, etc.

Schedule Generation Schemes

Given a sequence of activity, SGS determine the starting times of each activity

Serial schedule generation scheme (SSGS)

n stages, S_λ scheduled jobs, E_λ eligible jobs

Step 1 Select next from E_λ and schedule at earliest.

Step 2 Update E_λ and $R_k(\tau)$.
If E_λ is empty then STOP,
else go to Step 1.

Parallel schedule generation scheme (PSGS)

(Time sweep)

stage λ at time t_λ

S_λ (finished activities), A_λ (activities not yet finished), E_λ (eligible activities)

Step 1 In each stage select maximal resource-feasible subset of eligible activities in E_λ and schedule it at T_λ.

Step 2 Update E_λ, A_λ and $R_k(\tau)$.
If E_λ is empty then STOP,
else move to $t_{\lambda+1} = \min \left\{ \min_{j \in A_\lambda} C_j, \min_{i \in M} t_i^\mu \right\}$
and go to Step 1.

- If constant resource, it generates non-delay schedules
- Search space of PSGS is smaller than SSGS

Dispatching Rules

Determines the sequence of activities to pass to the schedule generation scheme

- activity based
- network based
- path based
- resource based

Static *vs* Dynamic

Local Search

All typical neighborhood operators can be used:

- Swap
- Interchange
- Insert

Genetic Algorithms

Recombination operator:

- One point crossover
- Two point crossover
- Uniform crossover

Outline

Reservations without slack – Interval Scheduling

Given:

- m parallel machines (resources)
- n activities
- r_j starting times (integers),
  d_j termination (integers),
  w_j or w_{ij} weight,
  M_j eligibility
- without slack $p_j = d_j - r_j$

Task: Maximize weight of assigned activities

Examples: Hotel room reservation, Car rental

Polynomially solvable cases

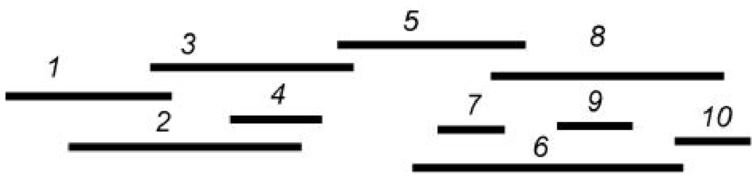
1. $p_j = 1$

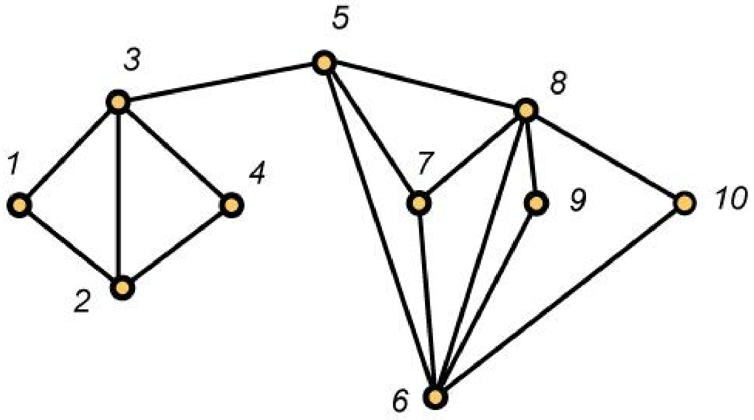
Solve an assignment problem at each time slot

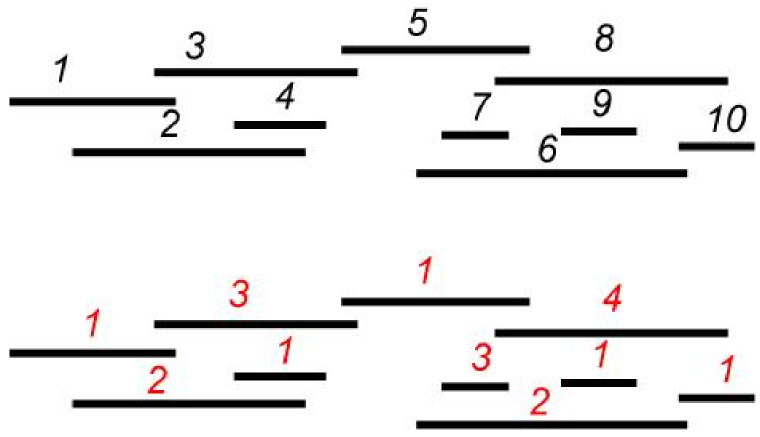
2. $w_j = 1$, $M_j = M$, Obj. minimize resources used

- Corresponds to coloring interval graphs with minimal number of colors
- Optimal greedy algorithm (First Fit):

  order $r_1 \leq r_2 \leq \ldots \leq r_n$
  Step 1 assign resource 1 to activity 1
  Step 2 **for** j from 2 to n **do**
  Assume k resources have been used.
  Assign activity j to the resource with minimum feasible value from $\{1, \ldots, k+1\}$







3. $w_j = 1$, $M_j = M$, Obj. maximize activities assigned

- Corresponds to coloring max # of vertices in interval graphs with k colors
- Optimal k-coloring of interval graphs:

order $r_1 \leq r_2 \leq \ldots \leq r_n$
$J = \emptyset$, $j = 1$

Step 1 if a resource is available at time r_j then assign activity j to that resource;
include j in J; go to Step 3

Step 2 Else, select j^* such that $C_{j^*} = \max_{j \in J} C_j$
if $C_j = r_j + p_j > C_{j^*}$ go to Step 3
else remove j^* from J, assign j in J

Step 3 **if** $j = n$ STOP **else** $j = j + 1$ go to Step 1

Outline

Reservations with Slack

Given:

- m parallel machines (resources)
- n activities
- r_j starting times (integers),
  d_j termination (integers),
  w_j or w_{ij} weight,
  M_j eligibility
- with slack $p_j \leq d_j - r_j$

Task: Maximize weight of assigned activities

Most constrained variable, least constraining value heuristic

$|M_j|$ indicates how much constrained an activity is

ν_{it}: # activities that can be assigned to i in $[t-1, t]$

Select activity j with smallest $I_j = f\left(\frac{w_j}{p_j}, |M_j|\right)$

Select resource i with smallest $g(\nu_{i,t+1}, \ldots, \nu_{i,t+p_j})$ (or discard j if no place free for j)

Examples for f and g:

$$f\left(\frac{w_j}{p_j}, |M_j|\right) = \frac{|M_j|}{w_j/p_j}$$

$$g(\nu_{i,t+1}, \ldots, \nu_{i,t+p_j}) = \max(\nu_{i,t+1}, \ldots, \nu_{i,t+p_j})$$

$$g(\nu_{i,t+1}, \ldots, \nu_{i,t+p_j}) = \sum_{l=1}^{p_j} \frac{\nu_{i,t+l}}{p_j}$$

Outline

Timetabling with Workforce or Personnel Constrains

There is only one type of operator that processes all the activities

Example:

- A contractor has to complete n activities.
- The duration of activity j is p_j
- Each activity requires a crew of size W_j.
- The activities are not subject to precedence constraints.
- The contractor has W workers at his disposal
- His objective is to complete all n activities in minimum time.

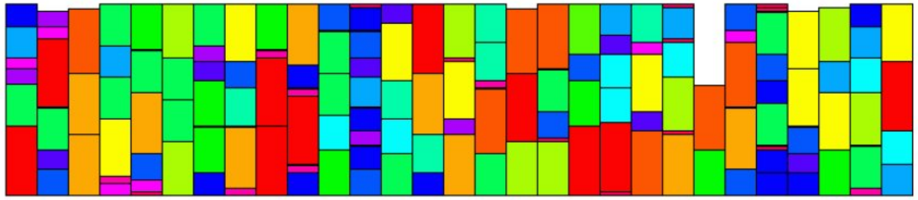
- RCPSP Model
- If p_j all the same ➜ Bin Packing Problem (still NP-hard)

Example: Exam scheduling

- Exams in a college with same duration.
- The exams have to be held in a gym with W seats.
- The enrollment in course j is W_j and
- all W_j students have to take the exam at the same time.
- The goal is to develop a timetable that schedules all n exams in minimum time.
- Each student has to attend a single exam.

- Bin Packing model
- In the more general (and realistic) case it is a RCPSP

Heuristics for Bin Packing



- Construction Heuristics
  - Best Fit Decreasing (BFD)
  - First Fit Decreasing (FFD) $C_{max}(FFD) \leq \frac{11}{9} C_{max}(OPT) + \frac{6}{9}$
- Local Search: [Alvim and Aloise and Glover and Ribeiro, 1999]

  Step 1: remove one bin and redistribute items by BFD

  Step 2: if infeasible, re-make feasible by redistributing items for pairs of bins, such that their total weights becomes equal (number partitioning problem)

[Levine and Ducatelle, 2004]

The solution before local search (the bin capacity is 10):
The bins: | 3 3 3 | 6 2 1 | 5 2 | 4 3 | 7 2 | 5 4 |

Open the two smallest bins:
Remaining: | 3 3 3 | 6 2 1 | 7 2 | 5 4 |
Free items: 5, 4, 3, 2

Try to replace 2 current items by 2 free items, 2 current by 1 free or 1 current by 1 free:
First bin: 3 3 3 → 3 5 2 new free: 4, 3, 3, 3
Second bin: 6 2 1 → 6 4 new free: 3, 3, 3, 2, 1
Third bin: 7 2 → 7 3 new free: 3, 3, 2, 2, 1
Fourth bin: 5 4 stays the same

Reinsert the free items using FFD:
Fourth bin: 5 4 → 5 4 1
Make new bin: 3 3 2 2
Final solution: | 3 5 2 | 6 4 | 7 3 | 5 4 1 | 3 3 2 2 |

Repeat the procedure: no further improvement possible

Outline

Timetabling with Different Operator or Tools

- There are several operators and activities can be done by an operator only if he is available
- Two activities that share an operator cannot be scheduled at the same time

Examples:

- aircraft repairs
- scheduling of meetings (people ➜ operators; resources ➜ rooms)
- exam scheduling (students may attend more than one exam ➜ operators)

If $p_j = 1$ ➜ Graph-Vertex Coloring (still NP-hard)

Mapping to Graph-Vertex Coloring

- activities ➜ vertices
- if 2 activities require the same operators ➜ edges
- time slots ➜ colors
- feasibility problem (if # time slots is fixed)
- optimization problem

DSATUR heuristic for Graph-Vertex Coloring

saturation degree: number of differently colored adjacent vertices

set of empty color classes $\{C_1, \ldots, C_k\}$, where $k = |V|$

Sort vertices in decreasing order of their degrees

Step 1 A vertex of maximal degree is inserted into C_1.

Step 2 The vertex with the maximal saturation degree is chosen and inserted according to the greedy heuristic (first feasible color). Ties are broken preferring vertices with the maximal number of adjacent, still uncolored vertices; if further ties remain, they are broken randomly.

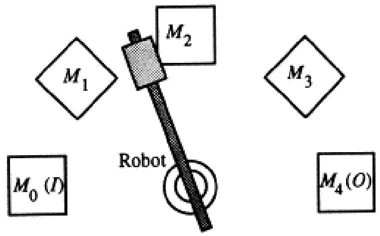
Outline

Resume: Job Shop

- Disjunctive graph representation [Roy and Sussman, 1964]
- Shifting Bottleneck Heuristic [Adams, Balas and Zawack, 1988]
- Local Search
- Generalizations:
  - Time lags d_{ij} to model:
    - set up times
    - synchronizations
    - deadlines
    - perishability (no-wait)
  - Blocking (alternative graph) ➜ Rollout

Exercise 1

Robotic Cell



Search for periodic pattern of moves (cycle)
one-unit cycle: the robot load (or unload) each machine exactly once
k-unit cycle: each activity is carried out exactly k times

Given:

- m machines $M_1, M_2, \ldots M_m$
- $c_{i,i+1}$ times of part transfer (unload+travel+load=activity) from M_i to M_{i+1}
- $d_{i,j}$ times of the empty robot from M_i to M_j ($c_{i,i+1} \geq d_{i,i+1}$)
- p_{ij} processing time of part j on machine i (identical vs different parts)

Task:

- Determine input time for each part t_j
- Minimize throughput $\rightsquigarrow$ minimize period

Alternative graph model with intermediate robot operations