**DM204**, 2010
SCHEDULING, TIMETABLING AND ROUTING

## Lecture 23
# Timetabling: Reservations and Education

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

# Outline

# Course Overview

- ✔ Problem Introduction
  - ✔ Scheduling classification
  - ✔ Scheduling complexity
  - ✔ RCPSP
- ✔ General Methods
  - ✔ Integer Programming
  - ✔ Constraint Programming
  - ✔ Heuristics
  - ✔ Dynamic Programming
  - ✔ Branch and Bound
- ✔ Scheduling Models
  - ✔ Single Machine
  - ✔ Parallel Machine and Flow Shop
  - ✔ Job Shop
  - ✔ Resource-Constrained Project Scheduling
- Timetabling
  - Reservations and Education
  - University Timetabling
  - Crew Scheduling
  - Public Transports
- Vechicle Routing
  - Capacited Models
  - Time Windows models
  - Rich Models

## Timetabling

- Educational Timetabling
  - School/Class timetabling
  - University timetabling
- Personnel/Employee timetabling
  - Crew scheduling
  - Crew rostering
- Transport Timetabling
- Sports Timetabling
- Communication Timetabling

# 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

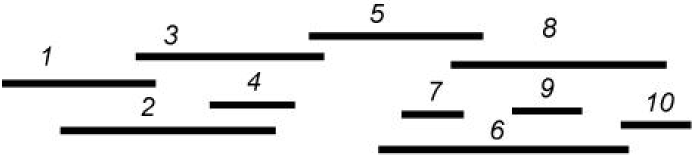
2. $w_j = 1$, $M_j = M$, Obj. minimize resources used

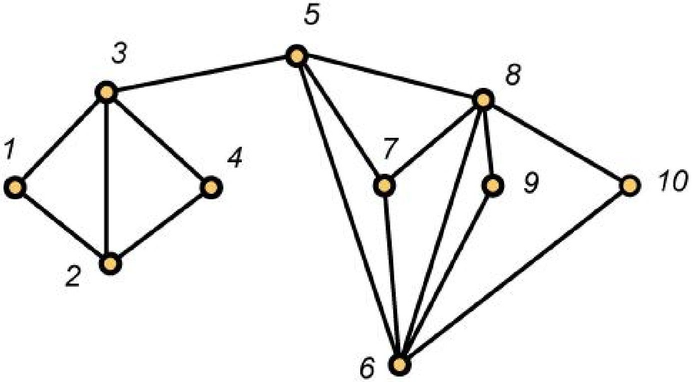
- Corresponds to coloring interval graphs with minimal number of colors
- Optimal greedy algorithm (First Fit):

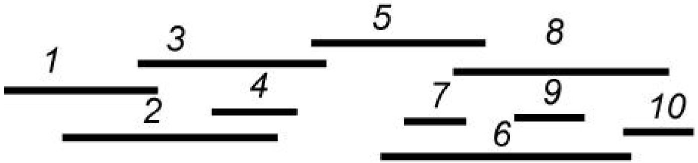
order $r_1 \leq r_2 \leq \ldots \leq r_n$

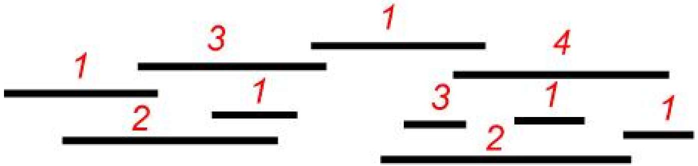
Step 1 assign resource 1 to activity 1

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 $\{1, \ldots, k+1\}$


5
3
8
1
4
7
9
10
2
6


5
3
8
1
7
4
9
10
2
6


1
2
3
4
5
6
7
8
9
10


1
2
3
1
1
2
3
4
1
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

# Heuristics

Most constrained variable, least constraining value heuristic

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

$\nu_{it}$: # 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 one Operator

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.

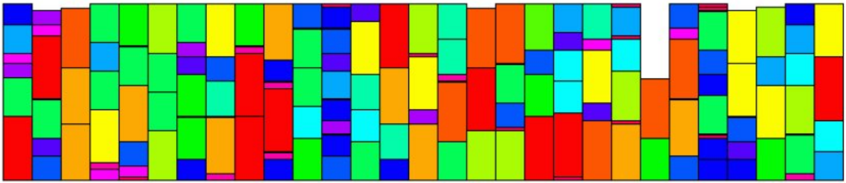
- RCPSP Model
- If $p_j$ all the same ➜ Bin Packing Problem (still NP-hard)

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 \| |

# Outline

# Timetabling with Operators

- 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

## Educational timetabling process

| | | | |
|---|---|---|---|
| **Phase:** | Planning | Scheduling | Dispatching |
| **Horizon:** | Long Term | Timetable Period | Day of Operation |
| **Objective:** | Service Level | Feasibility | Get it Done |
| **Steps:** | Manpower, Equipment | Weekly Timetabling | Repair |

# The Timetabling Activity

Assignment of events to a limited number of time periods and locations subject to constraints

Two categories of constraints:

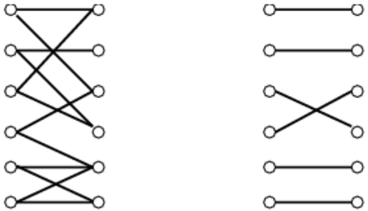
Hard constraints $H = \{H_1, \ldots, H_n\}$: must be strictly satisfied, no violation is allowed

Soft constraints $\Sigma = \{S_1, \ldots, S_m\}$: their violation should be minimized (determine quality)

Each institution may have some unique combination of hard constraints and take different views on what constitute the quality of a timetable.

## A recurrent sub-problem in Timetabling is Matching

**Input:** A (weighted) bipartite graph $G = (V, E)$ with bipartition $\{A, B\}$.
**Task**: Find the largest size set of edges $M \in E$ such that each vertex in $V$ is incident to at most one edge of $M$.



Efficient algorithms for constructing matchings are based on augmenting paths in graphs. An implementation is available at:
http://www.cs.sunysb.edu/~algorith/implement/bipm/implement.shtml

Theorem

**Theorem [Hall, 1935]:** *$G$ contains a matching of $A$ if and only if $|N(U)| \geq |U|$ for all $U \subseteq A$.*

# School Timetabling

[aka, teacher-class model]

The daily or weekly scheduling for all the classes of a high school, avoiding teachers meeting two classes in the same time.

**Input:**

- a set of classes $\mathcal{C} = \{C_1, \ldots, C_m\}$
  A class is a set of students who follow exactly the same program. Each class has a dedicated room.
- a set of teachers $\mathcal{P} = \{P_1, \ldots, P_n\}$
- a requirement matrix $\mathcal{R}_{m \times n}$ where $R_{ij}$ is the number of lectures given by teacher $R_j$ to class $C_i$.
- all lectures have the same duration (say one period)
- a set of time slots $\mathcal{T} = \{T_1, \ldots, T_p\}$ (the available periods in a day).

**Output:** An assignment of lectures to time slots such that no teacher or class is involved in more than one lecture at a time

IP formulation:

Binary variables: assignment of teacher $P_j$ to class $C_i$ in $T_k$

$$x_{ijk} = \{0, 1\} \quad \forall i = 1, \ldots, m;\ j = 1, \ldots, n;\ k = 1, \ldots, p$$

Constraints:

$$\sum_{k=1}^{p} x_{ijk} = R_{ij} \quad \forall i = 1, \ldots, m;\ j = 1, \ldots, n$$

$$\sum_{j=1}^{n} x_{ijk} \leq 1 \quad \forall i = 1, \ldots, m;\ k = 1, \ldots, p$$

$$\sum_{i=1}^{m} x_{ijk} \leq 1 \quad \forall j = 1, \ldots, n;\ k = 1, \ldots, p$$

## Graph model

Bipartite multigraph $G = (\mathcal{C}, \mathcal{P}, \mathcal{R})$:

- nodes $\mathcal{C}$ and $\mathcal{P}$: classes and teachers
- $R_{ij}$ parallel edges

Time slots are colors ➜ Graph-Edge Coloring problem

**Theorem: [König]** There exists a solution to (1) iff:

$$\sum_{i=1}^{m} R_{ij} \leq p \quad \forall j = 1, \ldots, n$$
$$\sum_{i=1}^{n} R_{ij} \leq p \quad \forall i = 1, \ldots, m$$

## Extension

From daily to weekly schedule
(timeslots represent days)

- $a_i$ max number of lectures for a class in a day
- $b_j$ max number of lectures for a teacher in a day

## IP formulation:

Variables: number of lectures to a class in a day

$$x_{ijk} \in N \quad \forall i = 1, \ldots, m;\ j = 1, \ldots, n;\ k = 1, \ldots, p$$

Constraints:

$$\sum_{k=1}^{p} x_{ijk} = R_{ij} \quad \forall i = 1, \ldots, m;\ j = 1, \ldots, n$$

$$\sum_{i=1}^{m} x_{ijk} \leq b_j \quad \forall j = 1, \ldots, n;\ k = 1, \ldots, p$$

$$\sum_{j=1}^{n} x_{ijk} \leq a_i \quad \forall i = 1, \ldots, m;\ k = 1, \ldots, p$$

## Graph model

Edge coloring model still valid but with

- no more than $a_i$ edges adjacent to $C_i$ have same colors and
- and more than $b_j$ edges adjacent to $T_j$ have same colors

**Theorem: [König]** There exists a solution to (2) iff:

$$\sum_{i=1}^{m} R_{ij} \leq b_j p \quad \forall j = 1, \ldots, n$$

$$\sum_{i=1}^{n} R_{ij} \leq a_i p \quad \forall i = 1, \ldots, m$$

- The edge coloring problem in the multigraph is solvable in polynomial time by solving a sequence of network flows problems $p$.
  Possible approach: solve the weekly timetable first and then the daily timetable

Further constraints that may arise:

- Preassignments
- Unavailabilities
  (can be expressed as preassignments with dummy class or teachers)

They make the problem NP-complete.

- Bipartite matchings can still help in developing heuristics, for example, for solving $x_{ijk}$ keeping any index fixed.

Further complications:

- Simultaneous lectures (eg, gymnastic)
- Subject issues (more teachers for a subject and more subject for a teacher)
- Room issues (use of special rooms)

So far feasibility problem.

Preferences (soft constraints) may be introduced

- Desirability of assignment $p_j$ to class $c_i$ in $t_k$

$$\min \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{p} d_{ijk} x_{ijk}$$

- Organizational costs: having a teacher available for possible temporary teaching posts

- Specific day off for a teacher

Introducing soft constraints the problem becomes a multiobjective problem.

Possible ways of dealing with multiple objectives:

- weighted sum
- lexicographic order
- minimize maximal cost
- distance from optimal or nadir point
- Pareto-frontier
- ...

# Heuristic Methods

Construction heuristic

Based on principles:

- most-constrained lecture on first (earliest) feasible timeslot
- most-constrained lecture on least constraining timeslot

Enhancements:

- limited backtracking
- local search optimization step after each assignment

More later

## Local Search Methods and Metaheuristics

High level strategy:

- Single stage (hard and soft constraints minimized simultaneously)
- Two stages (feasibility first and quality second)

Dealing with feasibility issue:

- partial assignment: do not permit violations of H but allow some lectures to remain unscheduled
- complete assignment: schedule all the lectures and seek to minimize H violations

More later