LOCAL SEARCH METHODS
APPLICATIONS AND ENGINEERING

Lecture 8

Timetabling and Scheduling

Marco Chiarandini

Outline

Outline

The Timetabling Activity

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

Types of timetabling

- Class Timetabling
- Exams Timetabling
- Courses Timetabling
- Employee Timetabling
- Crew Scheduling
- Crew Rostering
- Nurse Timetabling
- Transport Timetabling,
- Sport Timetabling,
- Communication Timetabling

In general timetabling varies from institution to institution in terms of constraints, requirements and their density

Two Types of Constraints:

- Hard constraints $\mathsf{H} = \{\mathsf{H}_1, \ldots, \mathsf{H}_\mathsf{n}\}$: their violation is not allowed
- Soft constraints $\Sigma = \{\mathsf{S}_1, \ldots, \mathsf{S}_\mathsf{m}\}$: their violation should be minimized

School (class/teacher, parent/teacher) model

A class is a set of students who follow exactly the same program.
Each class has a dedicated room.

Input: a set of classes $C = \{c_1, \ldots, c_m\}$, a set of teachers $T = \{t_1, \ldots, t_n\}$ and r periods $P_1, \ldots, P_r$. A requirement matrix $R_{m\times n}$ where r_{ij} is the number of lectures given by teacher t_j to class c_i.
Output: An assignment of lectures to periods such that no teacher is involved in more than one lecture at a time.

Graph model: **Edge Coloring** of a bipartite multigraph $G = (C, T, R)$.
Solvable in polynomial time by solving sequences of maximal matchings or network flows problems.

Further constraints:

- Weekly schedule, a period represents a set of periods, *e.g.*, the same period in each day of a week
- Maximum number of lectures in which C and T can be involved in a day (bounded edge coloring)
- Pre-assignments
- Unavailabilities
- Daily load is spread, balanced
- Specific day off for a teacher
- Desirability of assignment t_j to class c_i in p_k

Exam Model

There is one exam for each subject. Exams with common students are *conflicting* exams. There are constraints that tend to separate consecutive exams for students. The number of periods is not a strict constrain.

Input: A set of courses $C = \{c_1, \ldots, c_m\}$ and one exam for each course. A set of groups of exams $S = \{S_1, \ldots, S_p\}$ such that in each S_l there are students that take exams in S_l. A set of periods $P_1, \ldots, P_r$ and a set of rooms $R = \{R_1, \ldots, R_r\}$ available at each period.
Output: An assignment of each exam c_i to some period in such a way that no student is required to take more than one exam at a time.

Graph model: **Vertex Coloring** of a graph in which each exam c_i is a vertex and each conflict an edge.

Further constraints:

- Avoid that students have exams in consecutive periods
- Pre-assignments
- Unavailabilities
- Minimize the number of periods
- Room assignment with more than one exam per room

Course Model

Courses are made of lectures which have common students. Professors teach only one course. Assignment of rooms according to availability and suitability.

Input: A set of courses $C = \{C_1, \ldots, C_n\}$ each consisting of l_i lectures, $C_i = \{L_1, \ldots, L_{l_i}\}$. A set of curricula $S = \{S_1, \ldots S_r\}$ which are groups of courses with common students. A set of periods $P_1, \ldots, P_r$ and a set of rooms $R = \{R_1, \ldots, R_r\}$ available per each period.
Output: An assignment of each lecture L_i to some period in such a way that no student is required it take more than one lecture at a time.

Graph model: **Vertex Coloring** each lecture L_i constitutes a vertex and each conflict an edge.

An Example of Course Timetabling

2° anno - 1° Quadrimestre
dal 19 settembre 2005 al 25 novembre 2005

Fasce orarie	Lunedì	Martedì	Mercoledì	Giovedì	Venerdì
08:45 - 10:45	**Chimica e stato solido** Sergio Bruckner Aula: 51				
10:45 - 12:45	**Teoria delle reti elettriche** Fabrizio Bellina Aula: L			**Chimica e stato solido** Sergio Bruckner Aula: A	
12:45 - 14:45		**Teoria delle reti elettriche** Fabrizio Bellina Aula: F	**Controlli automatici I** Franco Blanchini Aula: F		**Controlli automatici I** Franco Blanchini Aula: F
14:45 - 16:45			**Teoria delle reti elettriche** Fabrizio Bellina Aula: F		**Chimica e stato solido** Sergio Bruckner Aula: A
16:45 - 18:45				**Controlli automatici I** Franco Blanchini Aula: F	

Il presente orario ha decorrenza dal giorno 26 settembre 2005

Typical Constraints in Course Timetabling

Hard Constraints:

- All required lectures must be scheduled
- Only one lecture is scheduled in one room at any period
- Lectures of the same curriculum must be scheduled in different periods
- Unavailabilities of professors in some periods
- Unavailabilities of rooms for courses due to lack of suitable facilities
- Pre-assignments of lectures in a determined room and period

Soft Constraints:

- Professors' preferences for teaching in some periods
- Preferences room-courses
- Room capacity must match the number of students attending a lecture
- Minimal number of working days (ex, a course with 5 lectures of 2 hours must be held in a minimum of 4 days)
- Intervals limiting the number of lectures per student in a day (ex, 2-3)
- Preferences on the minimal separation between consecutive lectures (ex, due to logistic constraints)
- Limit on the number of consecutive lectures
- Lectures of a same course must be scheduled in the same room

In Practice

A timetabling system consists of:

- Solver (written in a fast language, *i.e.*, C, C++) and various interfaces to handle the input and output
- Input and Output management (textual files or output in HTML format)
- Declaration of constraints (professors' preferences may be inserted directly through a web interface and stored in the information system of the University)

The whole timetabling process:

1. Collect data from the information system
2. Execute a few runs of the Solver starting from different solutions selecting the timetable of minimal cost. The whole computation time should not be longer than one night. This becomes a "draft" timetable.
3. The draft is shown to the professors who can require adjustments. The adjustments are obtained by defining new constraints to pass to the Solver.
4. Post-optimization of the "draft" timetable using the new constraints
5. The timetable can be further modified manually by using the Solver to validate the new timetables.

Solution Approaches

- Typically solved in two (or more) phases: first a feasible solution satisfying all hard constraints is found and then the soft constraints are considered.
  Issue: exiting or not from feasibility?
- Handling the soft constraints gives rise to a multi-objective problem. Three approaches:
  Combine objectives: combination of penalties into a single value by means of weights
  Goal programming: assign priorities and goal values to the objectives and optimize one objective at a time while imposing constraints on the others.
  Pareto-based: the whole Pareto-frontier of trade off solutions is determined
- The classroom assignment sub-problem can be solved efficiently if each period is considered independently

Search Space

Timetabling representation

- Graph coloring: it does not comprises the assignment of rooms
- Matrix representation: it comprises the assignment of rooms

$$
\text{Rooms}\quad
\begin{array}{c|cccccccccc}
 & \multicolumn{10}{c}{\text{Periods}} \\
 & P_1 & P_2 & & P_i & & P_j & & P_{45} \\
R_1 & -1 & L_4 & \cdots & \mathbf{L_{10}} & \cdots & L_{14} & \cdots & -1 \\
R_2 & L_1 & L_5 & \cdots & L_{11} & \cdots & \mathbf{L_{15}} & \cdots & -1 \\
R_3 & L_2 & L_6 & \cdots & \mathbf{L_{12}} & \cdots & -1 & \cdots & -1 \\
\vdots & \vdots & & \vdots & & \vdots & & \vdots & \\
R_r & L_3 & L_7 & \cdots & L_{13} & & L_{16} & \cdots & -1
\end{array}
$$

Two search approaches:

- Complete assignment of lectures (requirement constraints satisfied)
- Partial assignment of lectures (requirement constraints not satisfied)

Construction Heuristics

They are inspired by heuristics for GCP and consist of two main passages:

- Deciding the next lecture to schedule
- Deciding a place in the assignment matrix for the selected lecture

For example:

Step 1. Initialize the set $\widehat{L}$ of all unscheduled events with $\widehat{L} = L$.

Step 2. Choose a lecture $L_i \in \widehat{L}$ according to a *heuristic rule*.

Step 3. Let $\widehat{X}$ be the set of all positions for L_i in the assignment matrix with minimal violations of the hard constraints H.

Step 4. Let $\bar{X} \subseteq \widehat{X}$ be the subset of positions of $\widehat{X}$ with minimal violations of the soft constraints Σ is minimized.

Step 5. Choose an assignment for L_i in $\bar{X}$ according to a *heuristic rule*. Update information.

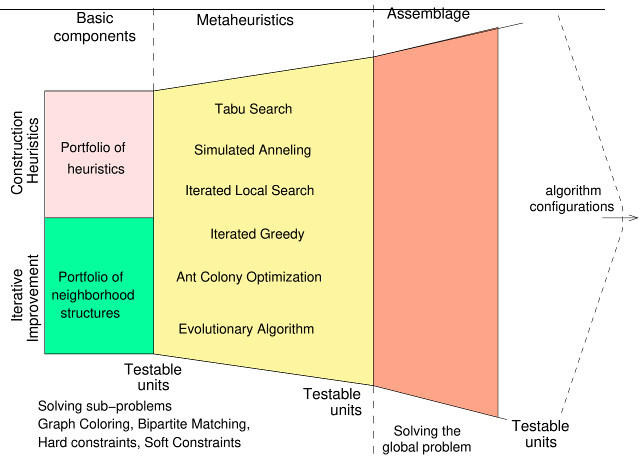
Step 6. Remove L_i from $\widehat{L}$, and go to step 2 until $\widehat{L}$ is not empty.

Neighborhood Structures for Perturbative Search

	Monday									Tuesday									Wednesday								
	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20	T21	T22	T23	T24	T25	T26	T27
R1	187	239	378	66	380	53	208	279		300	350	211	375	254	366	369	223	163	298		118	368	234	97	329	274	58
R2	360	345	2	153		354	91	61	319	349	278	86	204	316	220	323	176		314	7	108		50	312	235	330	
R3	263	71	186	67	222	288	99	24		237		232	253	117		195	203	102	207	287	290	146	286	358	303	277	
R4	181	160		90	82			193		206	156	152		341	179	171	226		4	348	127			365	213	80	
R5	324	291	309	339	267	283				269	170	299	311	34		65	216		275	199	26		27	327	33	39	285
R6	322	225	352	28	168	72	49	69	12	92	38	373	390	164	135	121	268	115	75	87	140	165	104	137	133	385	346
R7	228	31	107	371	30	355	46	227	246	271	182	313	224	128		89	258	356	343	280	35	109	306	43	83	11	154
R8	256	32	147	270	289	130	48	282		0	116	251	307	44	260	79	296		242	150	81	353	158	293	338	218	161
R9	396	144	173	78	25	183	387	337	240	132	328	212	370	308	336	244	126	14	231	51	342	136	93	129	266	393	155
R10	382	1	56	362	45	247	392	85	389	384	17	394	200		294	273	391	180	42	157	388	397	331	131	363	383	

- N_1: One-Exchange
- N_2: Two-Exchange
- N_3: Period Interchange
- N_4: Kempe Chain Interchange

Local Search Methods



An Example in Practice

Course Timetabling at the University of Edinburgh

Find an assignment of **lectures** to **periods** and **rooms** which is

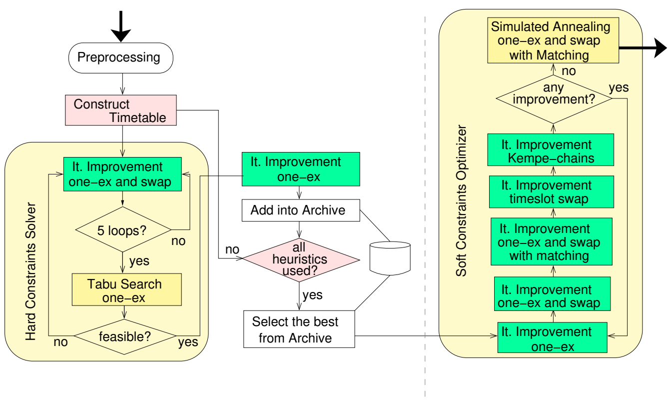
Feasible

rooms are only used by one lecture at a time,
each lecture is assigned to a suitable room,
no student has to attend more than one lecture at once; } Hard Constraints

and Good

not more than two lectures in a row for a student,
unpopular periods avoided (last in a day),
students do not have one single lecture in a day. } Soft Constraints

Algorithm Flowchart



Workforce Scheduling

Classification in terms of application areas:

- Crew Scheduling (in transportation and logistic)
- Nurse Scheduling (in health/social system)
- Employee Scheduling (in production and service)

Time horizon of the schedule, yielding typically decomposed problems:

- long term: "staffing" based on demand estimations
- short term: staff rostering
- real-time: reallocation

Rostering: the placing, subject to constraints, of persons into slots of a pattern. Often the resources will rotate through a roster of a determined duration (eg, one week).

The Terminology of *Nurse Rostering*

- Planning Period: the time (usually 4 weeks) over which the schedule take place
- Skill Category: qualification of the staff to accomplish a certain task
- Shift Type: well defined start and end time
- Coverage Constraints (= personnel demand/requirements): express the number of persons needed for every skill category and for every shift
- Time Related Constraints: restrictions or preferences on personal schedules
- Work Regulations: time related constraints due to the contract the personnel member has with the hospital

A List of Characteristics of *Nurse Rostering*

- Coverage and time related constraints considered as hard and/or soft constraints
- Flexibility of setting and defining problem parameters
  Fixed// Adaptable// User definable
- Cyclical and non-cyclical approaches
- Approaches for tackling coverage constraints
  Under-staffing Allowed/Not allowed// Over-staffing Allowed/Not allowed
- Number of skill categories and substitutability
  Skill categories scheduled separately// Hierarchical substitutability: people with a higher skill category can replace the lower// User definable substitutability
- Flexibility in setting and defining work regulations
  Identical work regulations for all people// Mixed workforce: Full time and half time personnel// User definable work regulations// Float personnel
- Different definitions of coverage constraints
  Constant// Weekdays-weekends// Fluctuating: the coverage constraints vary over the planning period
- Flexibility of defining shift types
  One single shift or no shifts defined: days// Three different shifts: usually referred to as Morning// Late and Night shift// Defined length: the intervals cannot be set by the users// User definable shifts: the number of different shifts// their start and end times and their length are set by the users
- Possible planning periods
  4 days// 1,2,3 weeks// 1 month

- Staff size
  User definable fixed number// To be minimized
- Time related constraints: Capacity
  Max number of assignments// Overtime// Max number of assignments to a particular Shift Type// Max number of shifts per week
- Time related constraints: Personal preferences
  Days off// Shifts off// Days on// Shift on// Shift patterns// Shift sequences
- Time related constraints: Consecutiveness constraints
  Max/min/fixed number of consecutive (free) days// Patterns// Free days after night shifts// No night shift before day off// Time between assignments// Consecutive shifts// Sequences of shift types// Mixture of day-night shifts per Week
- Time related constraints: Balance the workload
- Time related constraints: Weekends
  Complete weekends and extended weekends// Compensation of weekend work// Number of consecutive weekends
- Time related constraints: Others
  Preference days/nights// Changes in shifts on consecutive days// Maximum consecutive on/off/on patterns// People working together or not
- Objectives
  Minimize: violations of time related constraints// violations of coverage constraints// violations of coverage and time related constraints// number of employees// personnel cost// non-negative 'under' coverage// uneven distribution of shortages and surpluses over weekdays// deviation between scheduled nurses and demand // deviation between scheduled people and the total work capacity from the work regulations

Perturbative Search

Roster representation

		Shifts						
		Monday			Tuesday			$\cdots$
Personnel		Early	Day	Night	Early	Day	Night	$\cdots$
	P_1	$--$	T_1	$--$	$--$	T_3	$--$	$\cdots$
	P_2	T_1	$--$	$--$	T_1	$--$	$--$	$\cdots$
	P_3	T_2	$--$	$--$	$--$	$--$	T_3	$\cdots$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	P_r	T_3	$--$	$--$	T_3	$--$	T_1	$\cdots$

Neighborhood Structure

- 1-exchange within shifts
- 2-exchange within shifts
- insert/remove
- focused on specific constraints

Issues of Real Timetabling

- Fully address the needs and requirements of the real situation
- Generalized satisfaction: enterprise / employee
- Construction of the evaluation function, ie, in presence of conflicting objectives
- Robustness and flexibility, ie, minimizing the number of necessary real time changes consequent to unforeseen circumstances
- Maximize human/computer interaction
- Algorithm parameters hidden to the administrator
- Solution method hybridization
- Inter-disciplinarity: administration, management, psychology and software engineering

Definition Conflicting constraints

Outline

1. Timetabling
2. Scheduling

Problem Definition

Problem Definition

Given a set of **jobs** $\mathcal{J} = \{J_1, \dots, J_n\}$ that have to be processed by a set of **machines** $\mathcal{M} = \{M_1, \dots, M_m\}$ find a **schedule**, *i.e.*, a mapping of jobs to machines and processing times subject to feasibility and optimization constraints.

Problem Classification

Machine Environment $\boldsymbol{\alpha_1\alpha_2}|\beta_1\beta_2\beta_3\beta_4|\gamma_1\gamma_2$

- single machine/multi-machine ($1/m$)
- parallel machines: identical (P), uniform (Q), unrelated (R)
- Flow Shop (F), Open Shop (O), Job Shop and Group Shop (J)

Job Characteristics $\alpha_1\alpha_2|\boldsymbol{\beta_1\beta_2\beta_3\beta_4}|\gamma_1\gamma_2$

- single-stage/multi-stage $J_i = \{o_{i1}, \dots, o_{im(i)}\}$
- processing times of jobs or operations machine dependent/independent
- release r_i and due dates d_i associated with job J_i
- precedence constraints p_{ij} between jobs or operations
- weights w_i associated with J_i
- setup times t_{ij} for jobs J_i, J_j (machine dep/indep)

Objective $\alpha_1\alpha_2|\beta_1\beta_2\beta_3\beta_4|\boldsymbol{\gamma_1\gamma_2}$

Measures:

- Completion time C_i for job J_i
- Lateness $L_i = C_i - d_i$
- Tardiness $T_i = \max\{C_i - d_i, 0\}$
- Earliness $E_i = \max\{d_i - C_i, 0\}$

Typically, Minimize:

- maximum completion C_max = makespan
- $\sum w_i \cdot C_i$ total completion time
- $\sum w_i \cdot T_i$ total tardiness

Regular objective functions vs irregular

Single Machine Problems

- $1|d_j|L_{max}$, $1|d_j|T_{max}$ (Maximum Lateness/Tardiness) solved exactly by greedy **earliest due date**
- $1|d_j|L\sum T_i$ up to 500 jobs feasible exactly
- $1|d_j|L\sum w_iT_i$ up to 100 jobs feasible exactly

Flow Shop Problems

Characteristics:

- Multi-stage, Multi-machine
- The order in which the jobs pass through the machines is the same for all jobs
- Buffer policy:
  - FIFO $\Longrightarrow$ Permutation Flow Shop
  - change in sequence possible
- Buffer capacity:
  - infinite
  - limited $\Longrightarrow$ *blocking* and *no-wait requirements* issues

Permutation Flow Shop Problem (PFSP, $Fm||C_{max}$)

- m machines $\mathcal{M} = \{M_1, \dots, M_m\}$
- n jobs $\mathcal{J} = \{J_1, \dots, J_n\}$
- m operations per job $J_i = \{o_{i1}, \dots, o_{im}\}$
- 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}

Given a permutation π it is:

$$C_{max}(\pi) = \max_{1\leq t_1 \leq t_2 \leq \dots \leq t_{m-1} \leq n} \left(\sum_{j=1}^{t_1} p_{\pi(j)1} + \sum_{j=1}^{t_2} p_{\pi(j)2} + \dots + \sum_{j=1}^{t_m} p_{\pi(j)m} \right)$$

which can be computed by recursion.

Note:

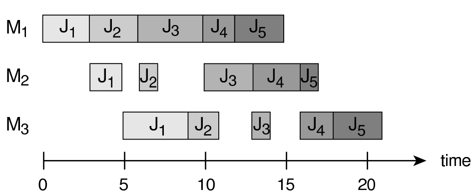
Single Machine and Permutation Flow Shop problems fall in the category of **permutation sequencing problems** which are all those problems whose solution can be simply represented by a permutation of jobs.

Example (PFSP)

Instance with 5 jobs and 3 machines (hence with 3 operations per job)

Jobs	J_1	J_2	J_3	J_4	J_5
p_{i1}	3	3	4	2	3
p_{i2}	2	1	3	3	1
p_{i3}	4	2	1	2	3

Gantt Chart of the canonical permutation $(J_1, J_2, J_3, J_4, J_5)$



Local Search Algorithms for the PFSP

- Construction Heuristics
  Johnson's rule, NEH ($\mathcal{O}(n^2m)$)
- Perturbative Search
  - Solution Representation
    Permutation of jobs
  - Neighborhood Structures
    Insert ($\mathcal{O}(n^2m)$), Interchange (Exchange), Swap (Transpose)
- Metaheuristics
  Stützle's Iterated Local Search, Novicki-Smutnicki's Tabu Search