

Materials on Constraint-based Scheduling (CBS)

Some topics are described in

- Philippe Baptiste, Philippe Laborie, Claude Le Pape, Wim Nuijten: Constraint-based Scheduling and Planning. Chapter 22 in Handbook of Constraint programming, pages 761-799, Elsevier, 2006.
- Roman Barták: Filtering Techniques in Planning and Scheduling, ICAPS 2006, June 6-10, 2006, Cumbria, England
<http://www.plg.inf.uc3m.es/icaps06/preprints/i06-tu2-allpapers.pdf>
- Philippe Baptiste, Claude Le Pape, Wim Nuijten: Constraint-based Scheduling, Kluwer Academic Publishers, 2001.

Constraint-based Scheduling: Introduction

- 1 Scheduling
- 2 CSP model
- 3 Resources
- 4 Optimization

Scheduling

- Scheduling

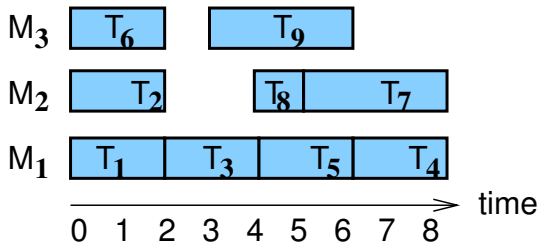
optimal resource allocation of a given set of activities in time

- resource or machine
- activity or task

- Machine $M_j, j = 1, \dots, 3$

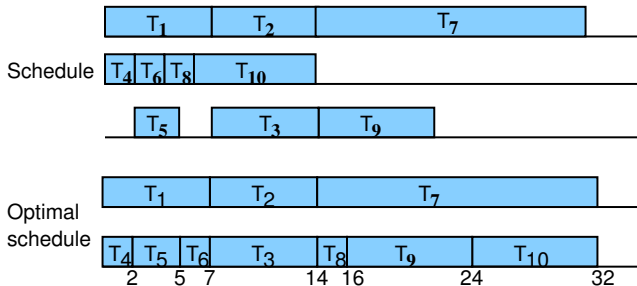
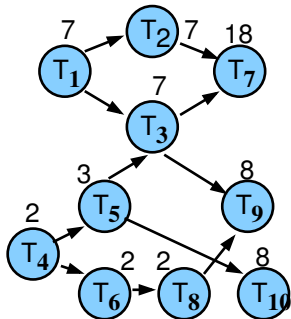
Task $T_i, i = 1, \dots, 9$

Machine-oriented Gantt chart



Example: Bicycle Assembly

- 3 workers who can perform tasks
- 10 tasks with its own duration
- Precedence constraints ($T_i \ll T_j$)
 - activity must be processed before other activity
- No preemption
 - activity cannot be interrupted during processing



Example: Classroom allocation

- One day seminar with several courses to be presented in several available rooms
- 8:00am – 4:00pm (periods 1,2,...,8)
- **14 courses** (A,B, ... N)
each course has several meetings with pre-assigned time periods
- **5 rooms** (1,2,3,4,5) ... resources
- **Find suitable room for each meeting**

Demands:	Course	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	Periods	2	8	4	1	3	6	7	2	1	5	6	3	8	2
				5	2	4		8	3	2		7	4		3
						5									4

Solution = Schedule/Timetable:	Periods	1	2	3	4	5	6	7	8
	Room 1	D	D		C	C	F		B
	Room 2	I	I	E	E	E		G	G
	Room 3		H	H		J	K	K	
	Room 4			N	N	N			M
	Room 5		A	L	L				

Scheduling problems

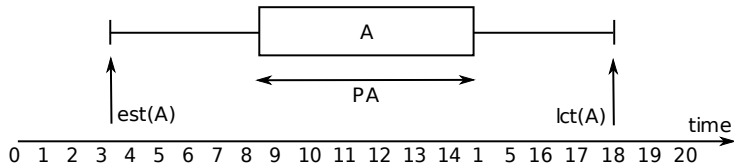
- **Project planning and scheduling**
 - software project planning
- **Machine scheduling**
 - allocation of jobs to computational resources
- **Scheduling of flexible assembly systems**
 - car production
- **Employees scheduling**
 - nurse rostering
- **Transport scheduling**
 - gate assignment for flights
- **Sports scheduling**
 - schedule for NHL
- **Educational timetabling**
 - timetables at school
- ...

Scheduling as a CSP: time assignment

Activity A is an entity occupying some space (resources) and time

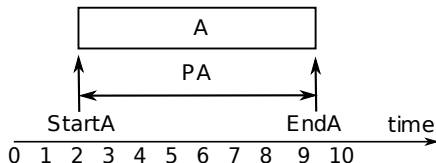
Variables and their domains for each activity for **time assignment**

- **StartA**: start time of the activity
 - activity cannot start before its **release date**
 - $est(A) = \min(\text{StartA})$, earliest start time
- **EndA**: completion time of the activity
 - activity must finish before the **deadline**
 - $lct(A) = \max(\text{EndA})$, latest completion time
- **PA**: processing time (duration) of the activity
 - $\text{StartA} = \{est(A), \dots, (lct(A)-PA)\}$
 - $\text{EndA} = \{(est(A)+PA), \dots, lct(A)\}$

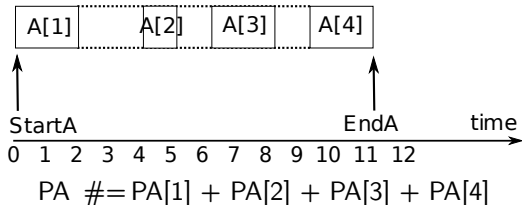


Scheduling as a CSP: basic constraints I.

- **Non-preemptive activity: no interruption** during processing
 - $\text{StartA} + \text{PA} \neq \text{EndA}$

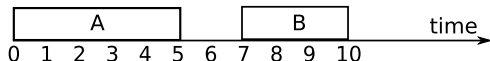


- **Preemptible activity: can be interrupted** during its processing
 - $\text{StartA} + \text{PA} \neq < \text{EndA}$



Scheduling as a CSP: basic constraints II.

- **Sequencing $A \ll B$** of activities A,B
(also: **precedence constraint** between activities A,B)
 - $\text{EndA} \neq < \text{StartB}$



- **Disjunctive constraint:** non-overlapping of activities A, B
 - non-preemptive activities
 - $A \ll B$ or $B \ll A$
 - $(\text{EndA} \neq < \text{StartB}) \neq \vee (\text{EndB} \neq < \text{StartA})$
 - related with the idea of unary resource

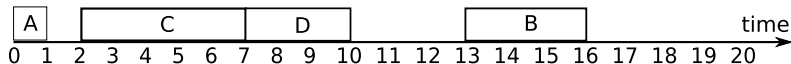
Scheduling as a CSP: resource allocation

Domain variables for resources

- **CapA**: requested capacity of the resource
 - unary resources
 - cumulative resources
 - producible/consumable resources
- **ResourceA**: alternative resources for A

Unary (disjunctive) resources

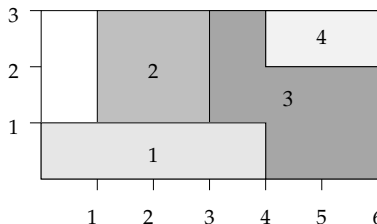
- Each activity requests unary capacity of the resource: $CapA=1$
- Single activity can be processed at given time
- Any two non-preemptive activities are related by the disjunctive constraint $A \ll B$ or $B \ll A$



- Example: one machine with jobs running on it
- `cumulative([task(StartA1,PA1,EndA1,1,A1),
..., task(StartAn,PAn,EndAn,1,An)],Options)`
 $A1, \dots, An$: activity identifiers
 $Options$: options for different propagation algorithms
4th parameter of the task = 1: unit consumption of the resource

Cumulative (discrete) resources

- Each activity uses some capacity of the resource **CapA**
- Several activities can be processed in parallel if a resource capacity is not exceeded



- Example: multi-processor computer with parallel jobs
- `cumulative([task(StartA1,PA1,EndA1,CapA1,A1), ...,`

`task(StartAn,PAn,EndAn,CapAn,An)], [limit(L)|Options])`

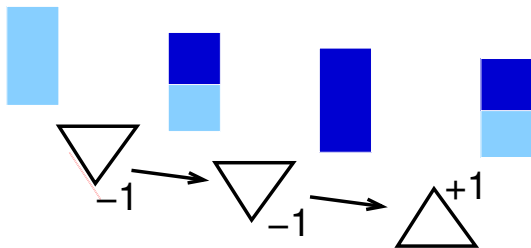
A1, ..., An: activity identifiers

limit(L): available capacity of the resource is **L**

Options: options for different propagation algorithms

Producible/consumable resources

- Resource = reservoir
- Activity consumes some quantity of the resource $CapA < 0$ or activity produces some quantity of the resource $CapA > 0$
- Minimal capacity is requested (consumption) and maximal capacity cannot be exceeded (production)



- Example: inventory for some products, activities producing them and activities using them in other production

Alternative resources

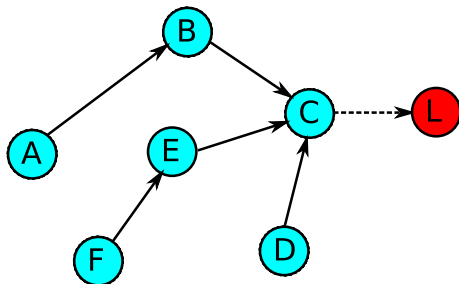
- Activity can be processed on a set of alternative resources
 - defined by the domain variable **ResourceA**
- One of them is selected for the activity
- Alternative unary resources
 - activity can be processed on any of the unary resources
 - can be modeled as one cumulative resource with resource capacity corresponding to the number of alternative unary resources
 - suitable for symmetric unary resources
- Example: any of the persons can process set of tasks

Optimization

Various criteria and objective function

Common criteria: **makespan**

- completion time of the last activity
- modeling
 - introduced a new additional activity L, $PL=0$
 - added precedence constraint for each activity T with no successor: $T \ll L$



- makespan = StartL