Introduction to Scheduling, Scheduling Algorithms

Jiří Vokřínek
A4M33PAH - 26.3.2012

Planning and Scheduling

- planning
  - problem: search for feasible set of actions fulfilling a goal
  - plan: partially ordered set of actions
  - actions: fully instantiated operators

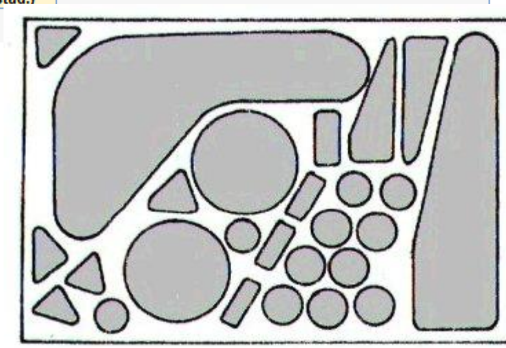
- scheduling:
  - problem: find an assignment of resources to actions
  - plan: sequence of resource-action assignment in time
  - can be modelled as parameters of an action
    - problem: planning algorithms tries out all possibilities (inefficient)
  - alternative approach:
    - allow unbound resource variables in plan (planning)
    - find assignment of resources to actions (scheduling)

Planning Techniques

- project planning
- Material Resource Planning (MRP)
- batch scheduling
- task ordering
- room scheduling
- notch planning

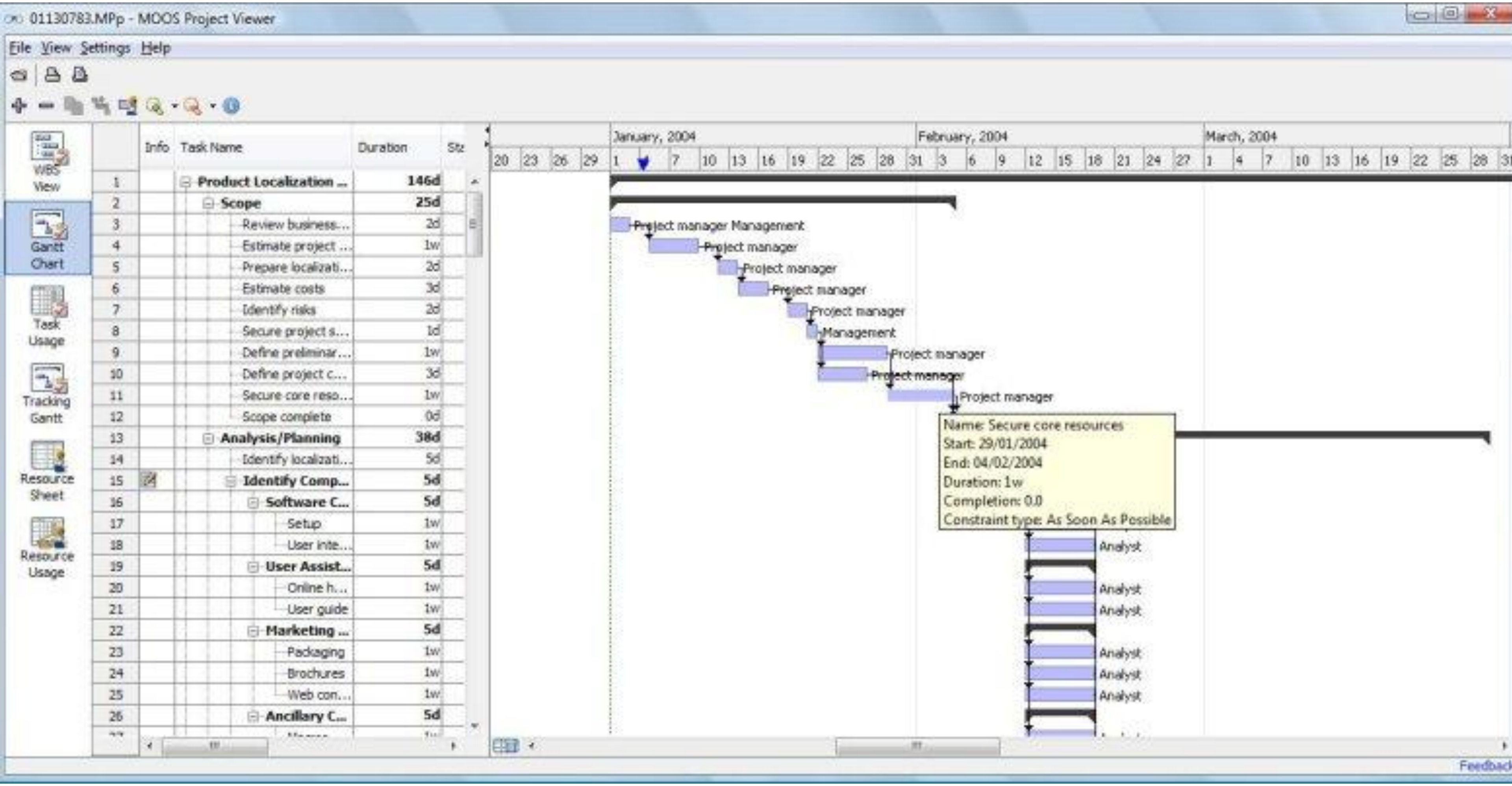
Rozvrh místnosti T2:C3-54 - sudý a lichý kalendářní týden

hodina	1, 2	3, 4	5, 6	7, 8	9, 10	11, 12	13, 14	15
čas	7:30 - 9:00	9:15 - 10:45	11:00 - 12:30	12:45 - 14:15	14:30 - 16:00	16:15 - 17:45	18:00 - 19:30	20:30
Pondělí		A2B99LES - Pře J. Hospodka, J. Dobeš 1 (29 stud.)		A2M34MIM - Pře M. Husák 1 (27 stud.)	A4M33BIA - Pře J. Kubalík, J. Drchal 1 (25 stud.)	A4M33PAH - Pře M. Pěchouček 1 (23 stud.)		
				A2M34MST - Pře M. Husák 1 (18 stud.)				
				X348MS - Pře M. Husák 1 (0 stud.)				



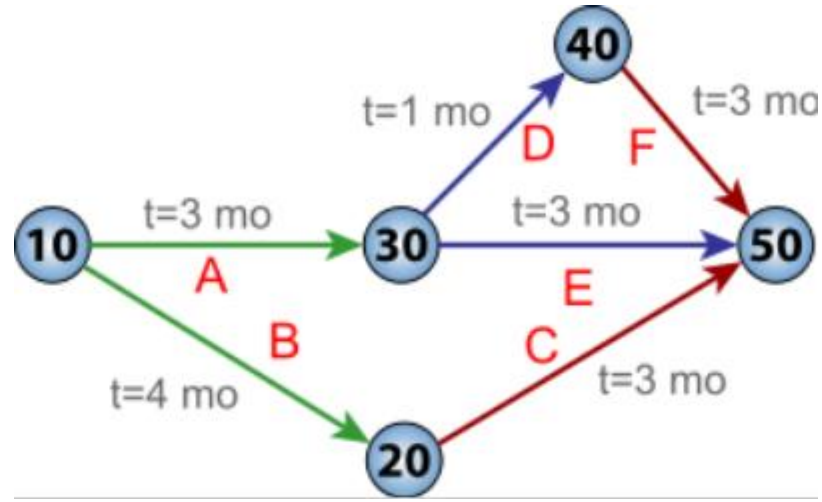
- project planning techniques:
  - Gantt charts
  - Program Evaluation and Review Technique
  - critical path analyses

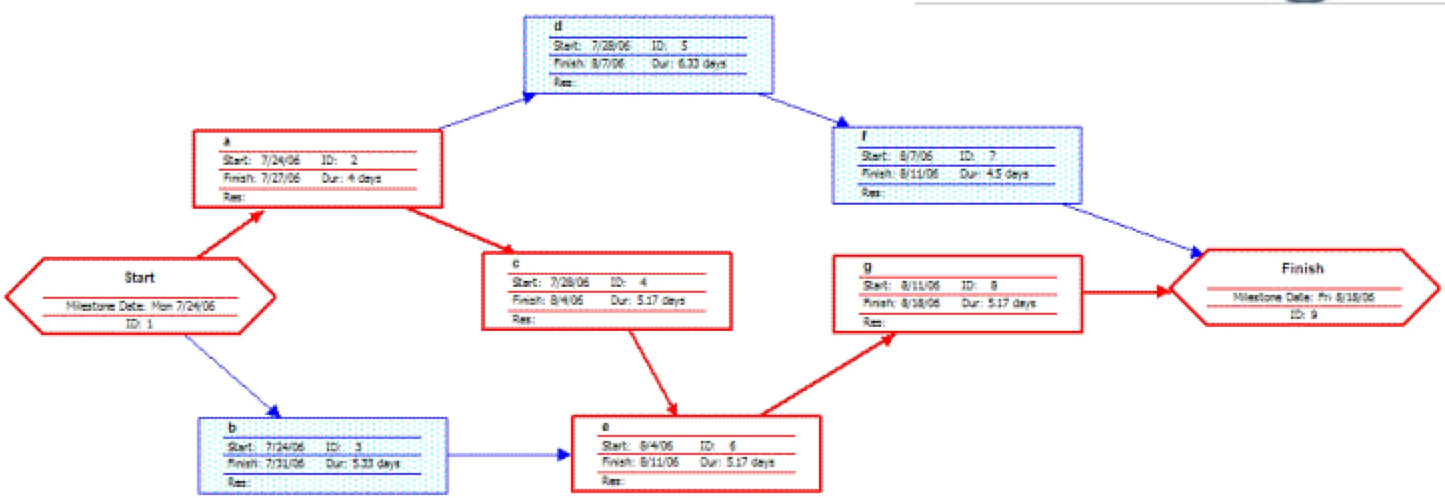
Gantt Chart

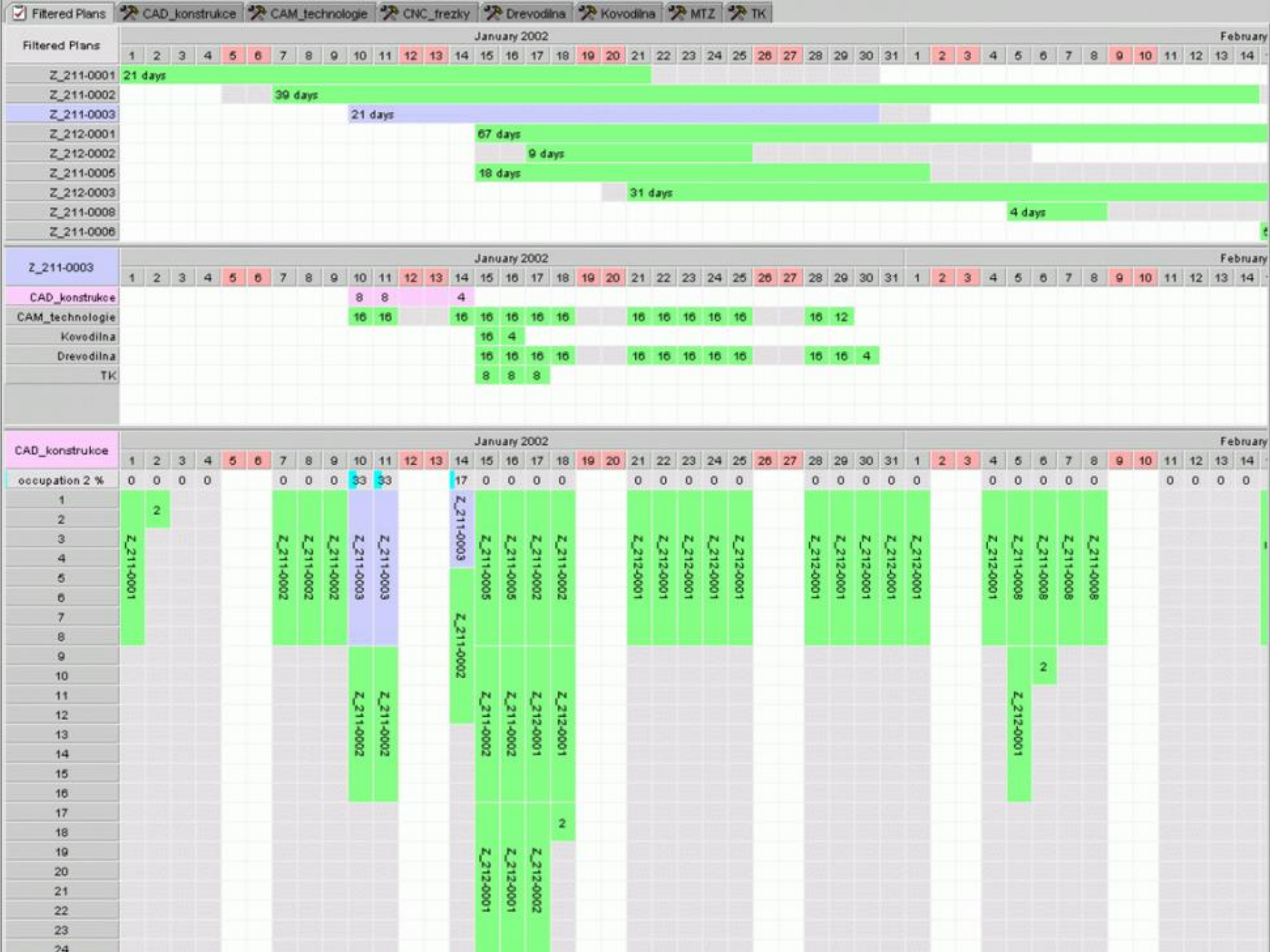


01130783.MPp - MOOS Project Viewer
File View Settings Help
WBS View
Gantt Chart
Task Usage
Tracking Gantt
Resource Sheet
Resource Usage
Info Task Name Duration
1 Product Localization ... 146d
2 Scope 25d
3 Review business... 2d
4 Estimate project ... 1w
5 Prepare localizati... 2d
6 Estimate costs 3d
7 Identify risks 2d
8 Secure project s... 1d
9 Define preliminar... 1w
10 Define project c... 3d
11 Secure core reso... 1w
12 Scope complete 0d
13 Analysis/Planning 38d
14 Identify localizati... 5d
15 Identify Comp... 5d
16 Software C... 5d
17 Setup 1w
18 User inte... 1w
19 User Assist... 5d
20 Online h... 1w
21 User guide 1w
22 Marketing ... 5d
23 Packaging 1w
24 Brochures 1w
25 Web con... 1w
26 Ancillary C... 5d
January, 2004
February, 2004
March, 2004
Project manager Management
Project manager
Management
Analyst
Name: Secure core resources
Start: 29/01/2004
End: 04/02/2004
Duration: 1w
Completion: 0.0
Constraint type: As Soon As Possible
Feedback

Program Evaluation and Review Technique (PERT)







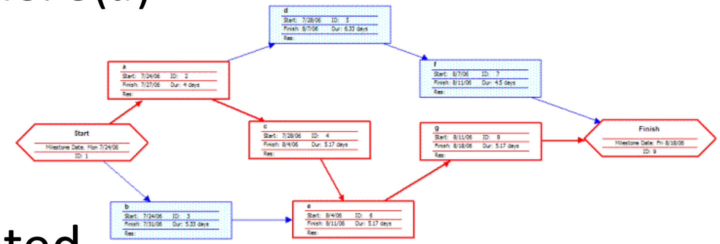
Filtered Plans
CAD_konstrukce
CAM_technologie
CNC_frezky
Drevodilna
Kovodilna
MTZ
TK
January 2002
February
Z_211-0001
21 days
Z_211-0002
39 days
Z_211-0003
21 days
Z_212-0001
67 days
Z_212-0002
9 days
Z_211-0005
18 days
Z_212-0003
31 days
Z_211-0008
4 days
Z_211-0006
Z_211-0003
CAD_konstrukce
CAM_technologie
Kovodilna
Drevodilna
TK
CAD_konstrukce
occupation 2 %

Actions and Resources

- resources: an entity needed to perform an action
  - state variables: modified by actions in absolute ways
    - example: move(*r,l,l'*):
    - location changes from *l* to *l'*
  - resource variables: modified by actions in relative ways
    - example: move(*r,l,l'*):
    - fuel level changes from *f* to *f-f'*

Actions with Time Constraints

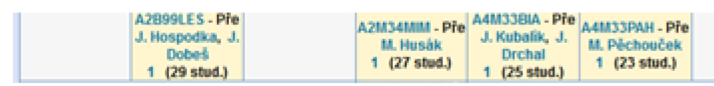
- Let *a* be an action in a planning domain:
  - attached time constraints:
    - earliest start time: $s_{min}(a)$ – actual start time: $s(a)$
    - latest end time: $s_{max}(a)$ – actual end time: $e(a)$
    - duration: $d(a)$



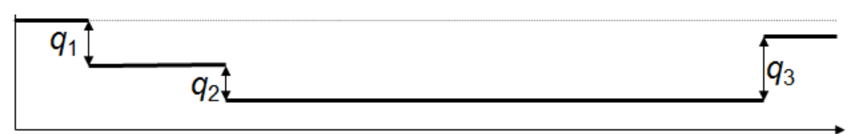
- action types:
  - preemptive actions: cannot be interrupted
    - $d(a) = e(a) - s(a)$
  - non-preemptive actions: can be interrupted
    - resources available to other actions during interruption

Actions with Resource Constraints

- Let *a* be an action in a planning domain:
  - attached resource constraints:
    - required resource: *r*
    - quantity of resource required: *q*
  - reusable: resource will be available to other actions after this action is completed

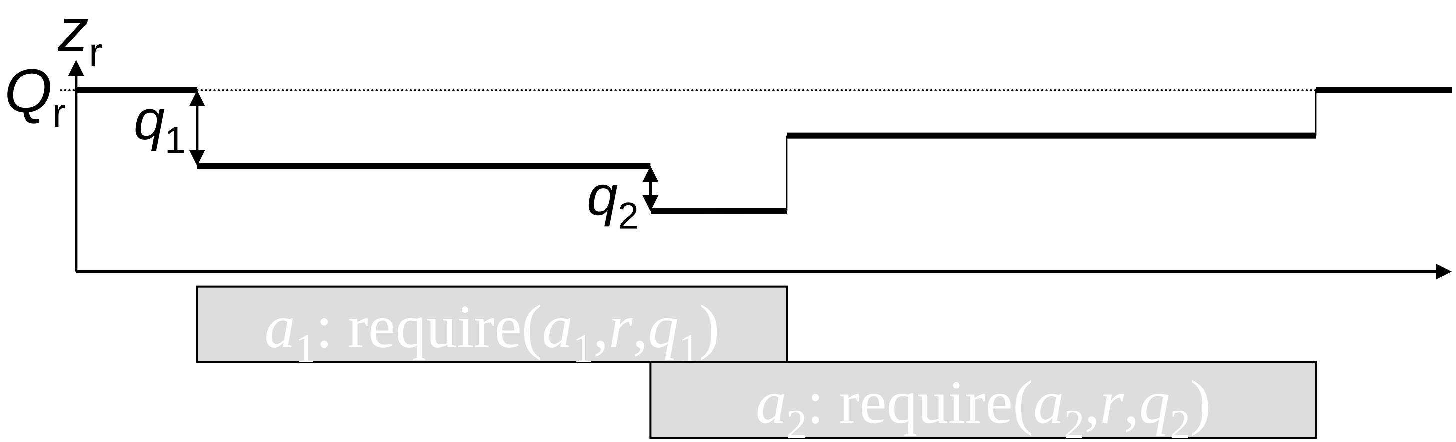


  - consumable: resource will be consumed when action is complete



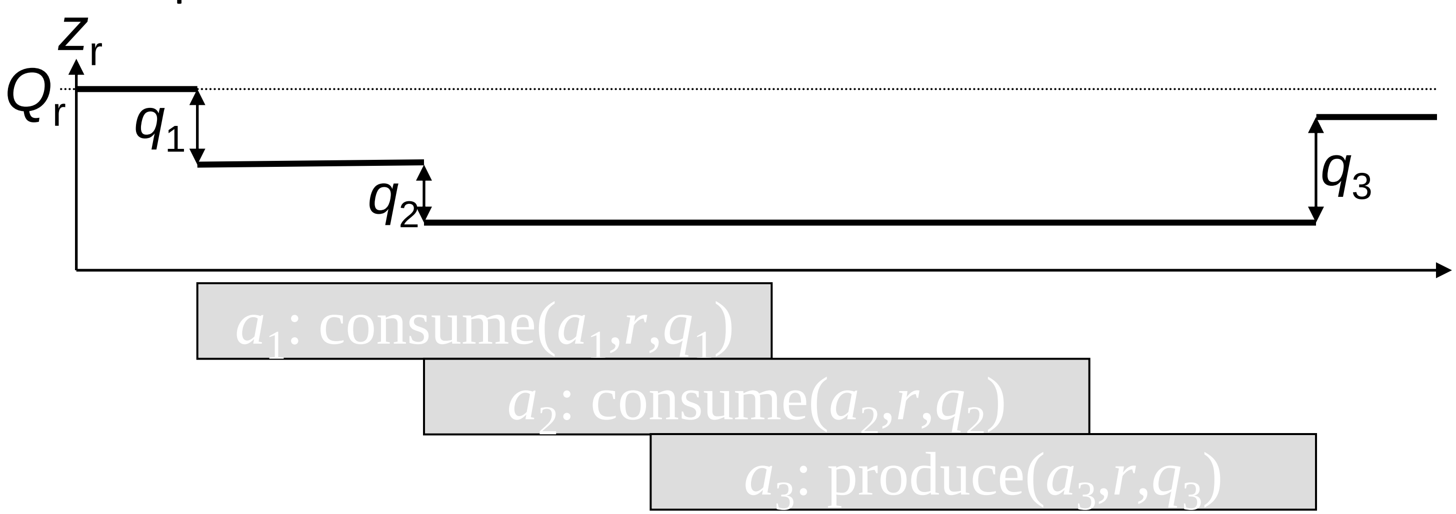
Reusable Resources

- resource availability:
  - total capacity: Q_r
  - current level at time t: $z_r(t)$
- resource requirements:
  - require(a,r,q): action a requires q units of resource r while it is active
- resource profile:



Consumable Resources

- resource availability:
  - total reservoir at t_0: Q_r
  - current level at time t: $z_r(t)$
- resource consumption/production:
  - consume(a,r,q): action a requires q units of resource r
  - produce(a,r,q): action a produces q units of resource r
- resource profile:



Other Resource Features

- discrete vs. continuous
  - countable number of units: cranes, bolts
  - real-valued amount: bandwidth, electricity
- unary
  - Q_r=1; exactly one resource of this type available
- sharable
  - can be used by several actions at the same time
- resources with states
  - actions may require resources in specific state

Combining Resource Constraints

- conjunction:
  - action uses multiple resources while being performed
- disjunction:
  - action requires resources as alternatives
  - cost/time may depend on resource used
- resource types:
  - resource-class(s) = $\{r_1,\ldots,r_m\}$: require(a,s,q)
  - equivalent to disjunction over identical resources

Cost Functions and Optimization Criteria

- cost function parameters
  - quantity of resource required
  - duration of requirement
- optimization criteria:
  - total schedule cost
  - makespan (end time of last action)
  - weighted completion time
  - (weighted) number of late actions
  - (weighted) maximum tardiness
  - resource usage

Planning vs. Scheduling

- Planning
  - feasibility of plan for *ONE* goal
  - duration (number of actions) in a plan

- Scheduling
  - utilization of resource(s) for *ALL* plans
  - total schedule cost or duration

- *It is hard to optimize both together …*

Machine Scheduling

- machine: resource of unit capacity
  - either available or not available at time *t*
  - cannot process two actions at the same time
- job *j*: partially ordered set of actions $a_{j1},\ldots,a_{jk}$
  - action a_{ji} requires
    - one resource type
    - for a number of time units
  - actions in same job must be processed sequentially
  - actions in different jobs are independent (not ordered)
- machine scheduling problem:
  - given: *n* jobs and *m* machines
  - schedule: mapping from actions to machines + start times

Material Resource Planning

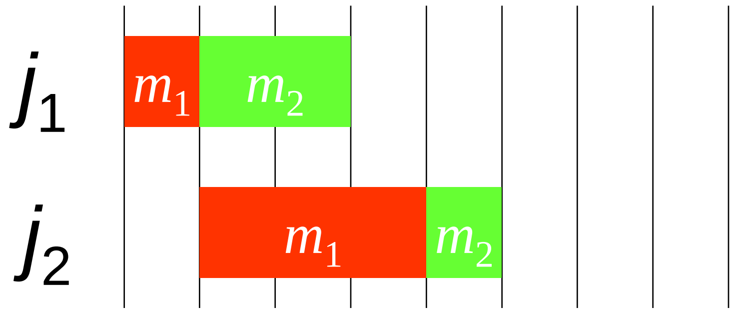
- machine: resource of countable capacity
  - available amount r_i at time t_i
  - can process any number of actions at the same time if $r_i >= 0$
- job j: partially ordered set of actions $a_{j1},\ldots,a_{jk}$
  - action a_{ji} requires
    - l resource types of q number each
    - for a number of time units
  - actions in same job must be processed sequentially
  - actions in different jobs are independent (not ordered)
- material resource planning problem:
  - given: n jobs and m machines
  - supply report: consumption of resources capacity by actions in time

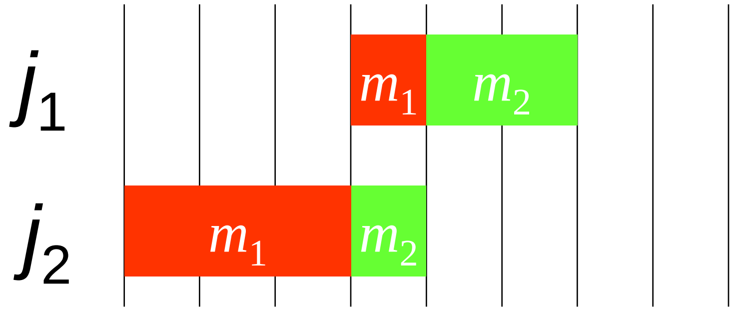
Example: Scheduling Problem

- machines:
  - m_1 of resource type r_1
  - m_2, m_3 of resource type r_2
- jobs:
  - j_1: $\langle r_1(3), r_2(3), r_1(3) \rangle$
    - three actions, totally ordered
    - a_{11} requires 3 units of resource type 1, etc.
  - j_2: $\langle r_2(3), r_1(5) \rangle$
  - j_3: $\langle r_1(3), r_1(2), r_2(3), r_1(5) \rangle$

Example: Schedules by Job

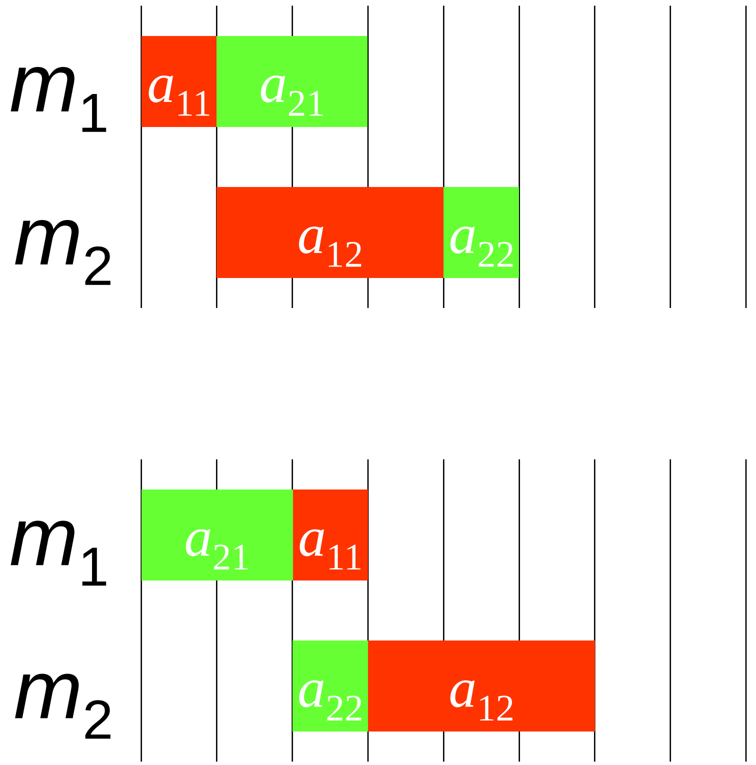
- machines:
  - m_1 of type r_1
  - m_2 of type r_2
- jobs:
  - j_1: $\langle r_1(1), r_2(2) \rangle$
  - j_2: $\langle r_1(3), r_2(1) \rangle$





Example: Schedules by Machine

- machines:
  - m_1 of type r_1
  - m_2 of type r_2
- jobs:
  - j_1: $\langle r_1(1), r_2(2)\rangle$
  - j_2: $\langle r_1(3), r_2(1)\rangle$



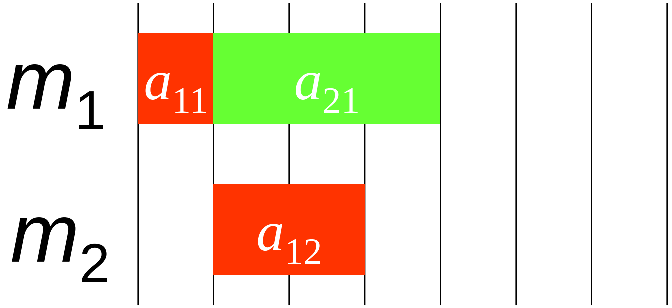
Assignable Actions

- Let P be a machine scheduling problem. Let S be a partially defined schedule.
- An action a_{ji} of some job j_l in P is <u>unassigned</u> if it does not appear in S.
- An action a_{ji} of some job j_l in P is <u>assignable</u> if it has no unassigned predecessors in S.

Example: Assignable Actions

- problem P:
  - machines:
    - m_1 of type r_1
    - m_2 of type r_2
  - jobs:
    - j_1: $\langle r_1(1), r_2(2) \rangle$
    - j_2: $\langle r_1(3), r_2(1) \rangle$
    - j_3: $\langle r_1(3), r_2(1), r_1(3) \rangle$

partial schedule S:



- unassigned:
  - a_{22}, a_{31}, a_{32}, a_{33}
- assignable:
  - a_{22}, a_{31}

Earliest Assignable Time

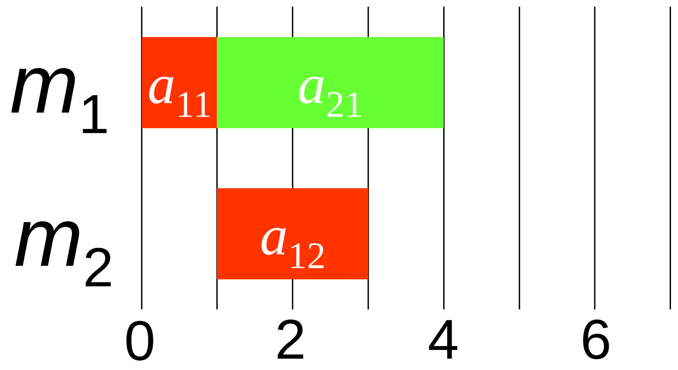
- Let a_{ji} be an assignable action in S. The <u>earliest assignable time</u> for a_{ji} on machine m in S is:
  - the end of the last action currently scheduled on m in S, or
  - the end of the last predecessor (a_{j0} … $a_{ji\text{-}1}$) in S, or
  - the earliest start time $s_{min}(a_{ji})$,

  whichever comes later.

Example: Earliest Assignable Time

- problem P (R2|prec|C_max):
  - machines:
    - m_1 of type r_1
    - m_2 of type r_2
  - jobs:
    - j_1: $\langle r_1(1), r_2(2)\rangle$
    - j_2: $\langle r_1(3), r_2(1)\rangle$
    - j_3: $\langle r_1(3), r_2(1), r_1(3)\rangle$

partial schedule S:



- earliest assignable time for a_{22} on m_2: 4
- earliest assignable time for a_{31} on m_1:4

Heuristic Search

heuristicScheduler(*P*,*S*)

assignables ← *P*.getAssignables(*S*)

if *assignables*.isEmpty() **then return** *S*

action ← *assignables*.selectOne()

machines ← *P*.getMachines(*action*)

machine ← *machines*.selectOne()

time ← *S*.getEarliestAssignableTime(*action*, *machine*)

S ← S + assign(*action*, *machine*, *time*)

return heuristicScheduler(*P*,*S*)

Scheduling Algorithms

- First In, First Out (FIFO) known also as First Come, First Served (FCFS)
- Last In, First Out (LIFO)
- Shortest Remaining Time First (SRTF), Shortest Job First (SJF)
- priority ordering
- Round-robin (RR) scheduling
- critical path priority ordering

Scheduling Algorithms

- scheduling problem $\boldsymbol{\alpha|\beta|\gamma}$
- α – machine environment: **1** (single machine), **P*m*** (*m* identical machines), **Q*m*** (as P with different speeds), **R*m*** (as P, but unrelated)
- β – problem specs: $\boldsymbol{r_i}$ (release time), $\boldsymbol{d_i}$ (deadline), **pmtn** (preemptive), $\boldsymbol{size_i}$ (multi-machine), **prec** (precedences), …
- γ – objective function: $\boldsymbol{C_{\max}}$, $\boldsymbol{L_{\max}}$, $\boldsymbol{E_{\max}}$, $\boldsymbol{T_{\max}}$, $\sum C_i$, $\sum L_i$, $\sum E_i$, $\sum T_i$,

the scheduling zoo: http://www-desir.lip6.fr/~durrc/query/

Example: FCFS

- First In, First Out (FIFO) known also as First Come, First Served (FCFS)
- problem – average waiting time depends on arrival order
- advantage – simple algorithm

Example: LIFO

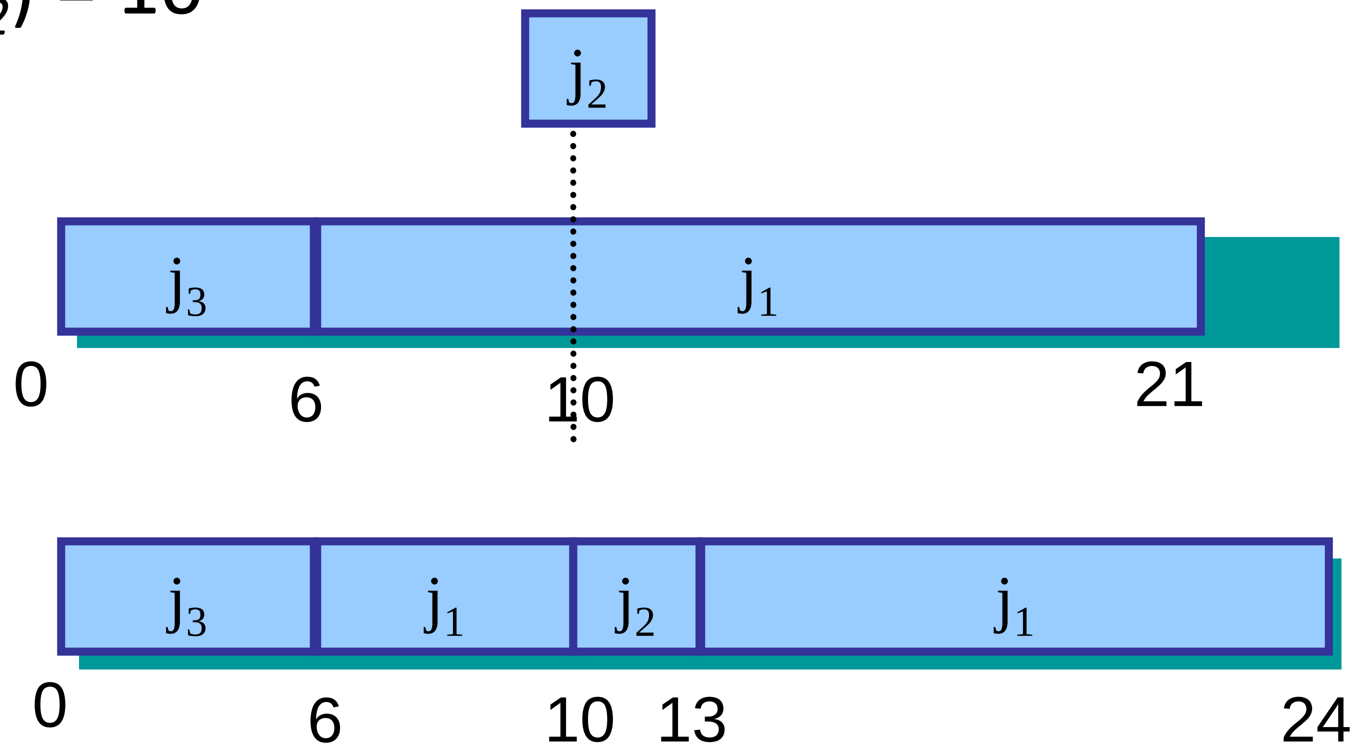
- Last In, First Out (LIFO)
- problem – early processes may never be served (for dynamic scheduling)
- advantage – newly arrived jobs have low response times

Example: SJF

- Shortest Job First (SJF)
- provably optimal for minimizing average waiting time

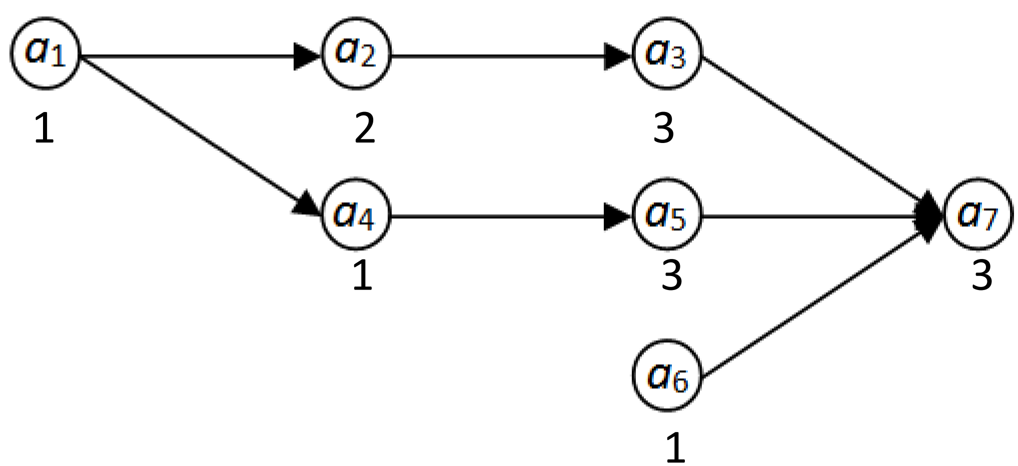
Example: SRTF

- Shortest Remaining Time First (SRTF)
- preemptive variant of SJF
- $s_{min}(j_2) = 10$



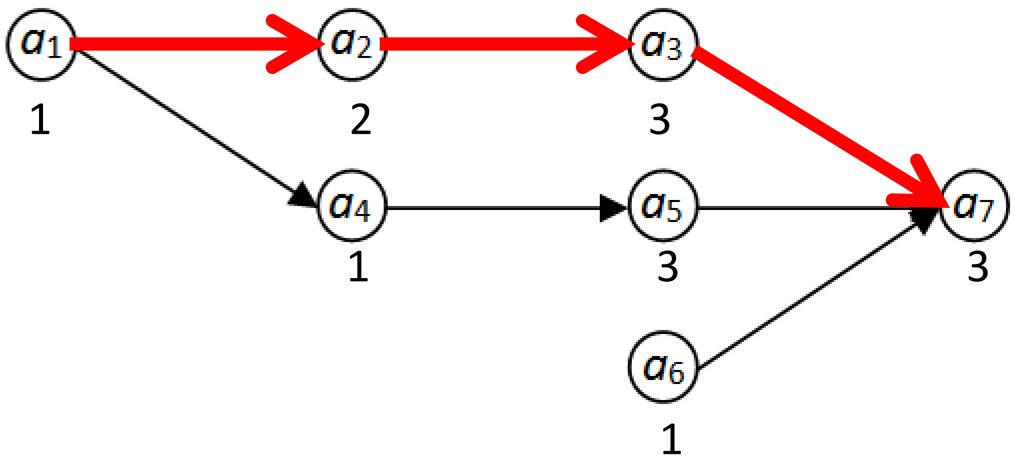
Example: critical path

- problem P (P|prec|C_max):
  - job:
    - j: $\langle a_1(1), a_2(2), a_3(3), a_4(1), a_5(3), a_6(1), a_7(3)\rangle$
    - $a_1<a_2$, $a_2<a_3$, $a_1<a_4$, $a_4<a_5$, $a_5<a_7$, $a_6<a_7$, $a_3<a_7$



Example: critical path

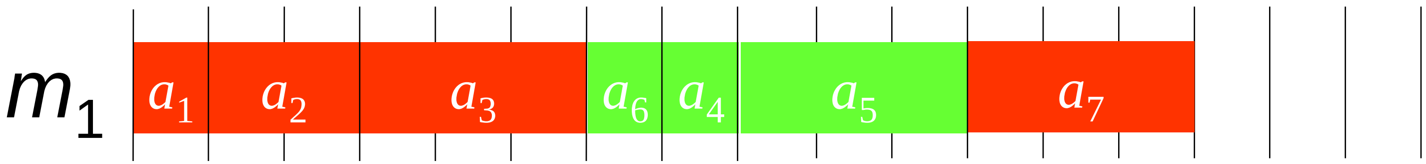
- problem P (P|prec|C_max):
  - job:
    - j: $\langle a_1(1), a_2(2), a_3(3), a_4(1), a_5(3), a_6(1), a_7(3)\rangle$
    - $a_1<a_2$, $a_2<a_3$, $a_1<a_4$, $a_4<a_5$, $a_5<a_7$, $a_6<a_7$, $a_3<a_7$



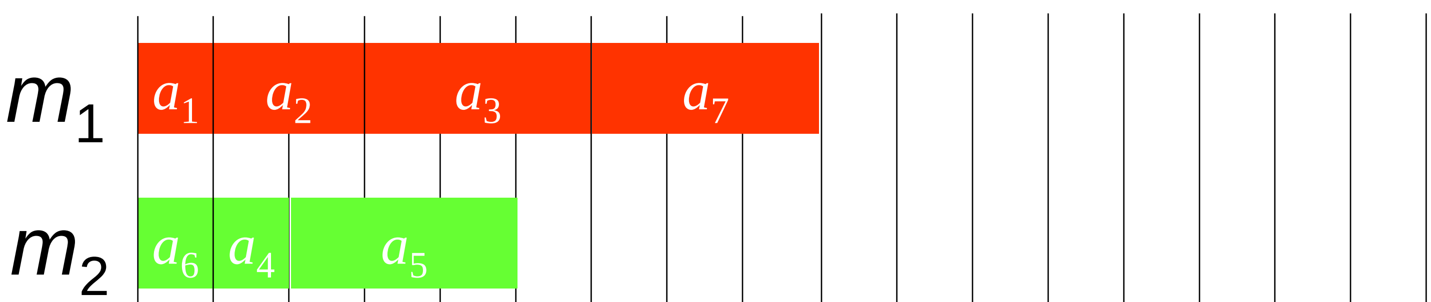
critical path length: 9

Example: critical path

- problem *P* (1|prec|C_max, P2|prec|C_max):
  - job:
    - j: $\langle a_1(1), a_2(2), a_3(3), a_4(1), a_5(3), a_6(1), a_7(3)\rangle$
    - $a_1<a_2$, $a_2<a_3$, $a_1<a_4$, $a_4<a_5$, $a_5<a_7$, $a_6<a_7$, $a_3<a_7$
  - machines: m_1 of one type (upper-bound schedule length = 14)



  - machines: m_1, m_2 of the same type
  - (with unlimited machines: lower-bound schedule length = 9)



Literature

- Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning – Theory and Practice*, chapter 15. Elsevier/Morgan Kaufmann, 2004.
- Michael Pinedo. *Scheduling: Theory, Algorithms and Systems*, Prentice Hall, 2001.
- Peter Brucker. *Scheduling Algorithms*, Springer Verlag, 2004.