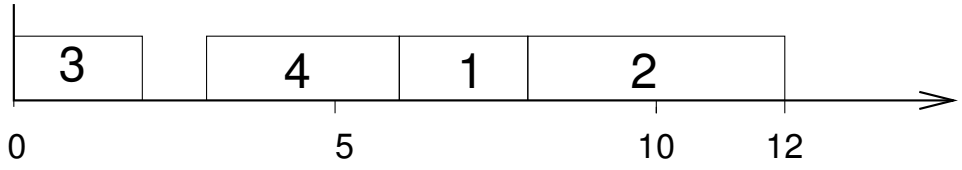
- $1|r_j|C_{max}$
  - given: n jobs with processing times $p_1, \ldots, p_n$ and release dates $r_1, \ldots, r_n$
  - jobs have to be scheduled without preemption on one machine taking into account the earliest starting times of the jobs, such that the makespan is minimized
  - $n = 4$, $p = (2, 4, 2, 3)$, $r = (5, 4, 0, 3)$

Classification - Examples

- $1|r_j|C_{max}$
  - given: n jobs with processing times $p_1, \ldots, p_n$ and release dates $r_1, \ldots, r_n$
  - jobs have to be scheduled without preemption on one machine taking into account the earliest starting times of the jobs, such that the makespan is minimized
  - $n = 4$, $p = (2, 4, 2, 3)$, $r = (5, 4, 0, 3)$

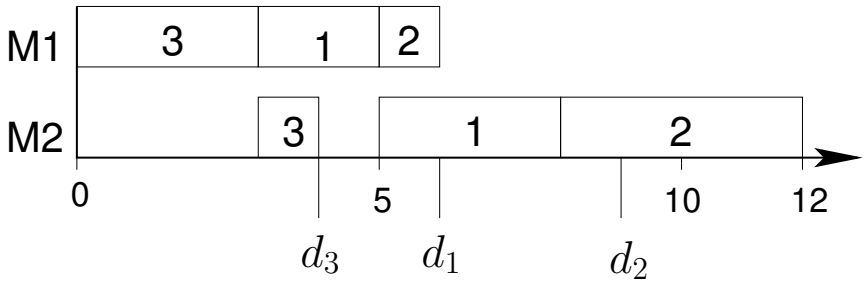


Feasible Schedule with $C_{max} = 12$ (schedule is optimal)

- $F2||\sum w_jT_j$
  - given n jobs with weights $w_1, \ldots, w_n$ and due dates $d_1, \ldots, d_n$
  - operations (i, j) with processing times $p_{ij},\ i = 1, 2;\ j = 1, \ldots, n$
  - jobs have to be scheduled on two machines such that operation $(2, j)$ is scheduled on machine 2 and does not start before operation $(1, j)$, which is scheduled on machine 1, is finished and the total weighted tardiness is minimized
  - $n = 3,\ p = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 4 & 1 \end{pmatrix},\ w = (3, 1, 5),\ d = (6, 9, 4)$

Classification - Examples

- $F2||\sum w_jT_j$
  - given n jobs with weights $w_1, \ldots, w_n$ and due dates $d_1, \ldots, d_n$
  - operations (i, j) with processing times $p_{ij},\ i = 1, 2;\ j = 1, \ldots, n$
  - jobs have to be scheduled on two machines such that operation $(2, j)$ is schedules on machine 2 and does not start before operation $(1, j)$, which is scheduled on machine 1, is finished and the total weighted tardiness is minimized
  - $n = 3,\ p = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 4 & 1 \end{pmatrix},\ w = (3, 1, 5),\ d = (6, 9, 4)$



$$\sum w_jT_j = 3(8-6) + (12-9) + 5(4-4) = 9$$

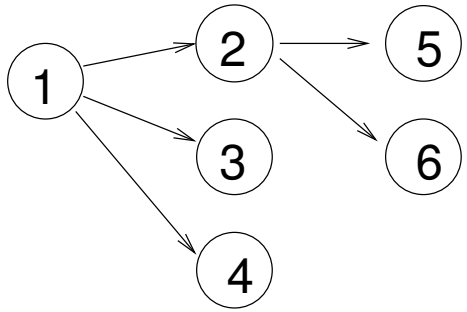
Classes of Schedules

- Nondelay Schedules:
  A feasible schedule is called a nondelay schedule if no machine is kept idle while a job/an operation is waiting for processing

  Example: $P3|prec|C_{max}$

  $n = 6$

  $p = (1, 1, 2, 2, 3, 3)$



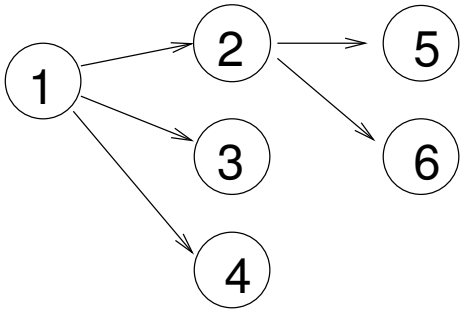
Classes of Schedules

- Nondelay Schedules:
  A feasible schedule is called a nondelay schedule if no machine is kept idle while a job/an operation is waiting for processing

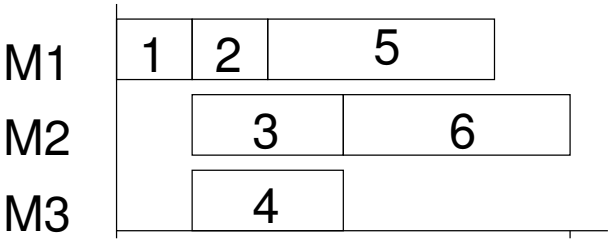
  Example: $P3|prec|C_{max}$

  $n = 6$

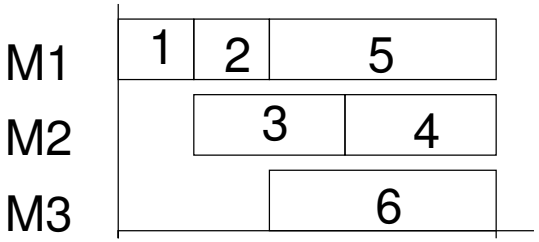
  $p = (1, 1, 2, 2, 3, 3)$



Best nondelay:



Optimal



Classes of Schedules

Remark: restricted to non preemptive schedules

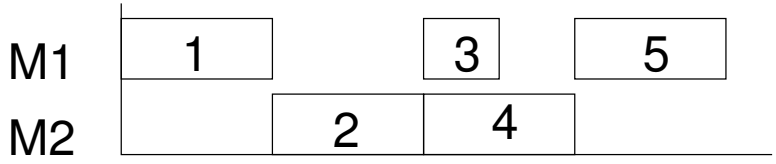
- Active Schedules:
  A feasible schedule is called active if it is not possible to construct another schedule having at least one job/operation finishing earlier and no job/operation finishing later, possibly by changing the order of processing on the machines and
- Semi-Active Schedules:
  A feasible schedule is called semi-active if no job/operation can be finishing earlier without changing the order of processing on any one of the machines.

Classes of Schedules

Examples of (semi)-active schedules:

Prec: $1 \rightarrow 2;\ 2 \rightarrow 3$

not semi-active

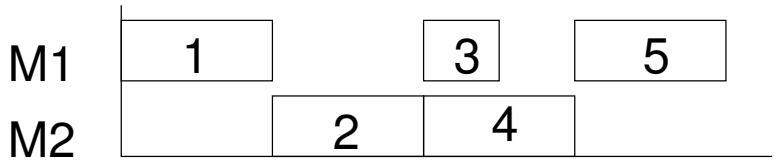


Classes of Schedules

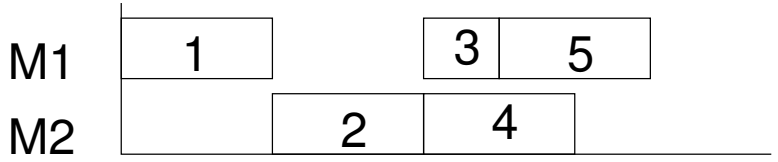
Examples of (semi)-active schedules:

Prec: $1 \rightarrow 2;\ 2 \rightarrow 3$

not semi-active



semi-active

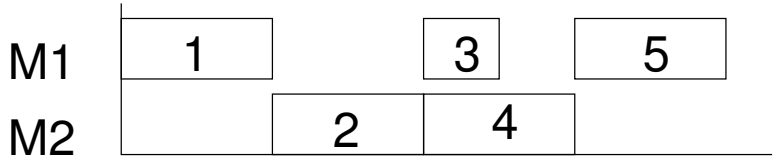


Classes of Schedules

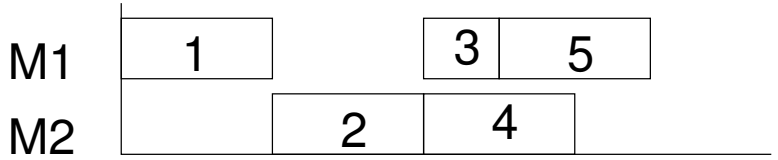
Examples of (semi)-active schedules:

Prec: $1 \rightarrow 2;\ 2 \rightarrow 3$

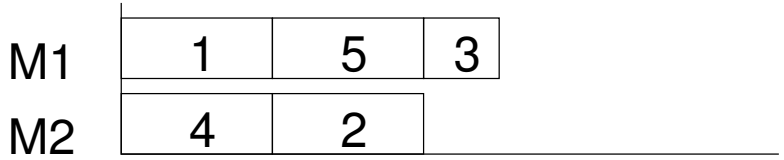
not semi-active



semi-active



active

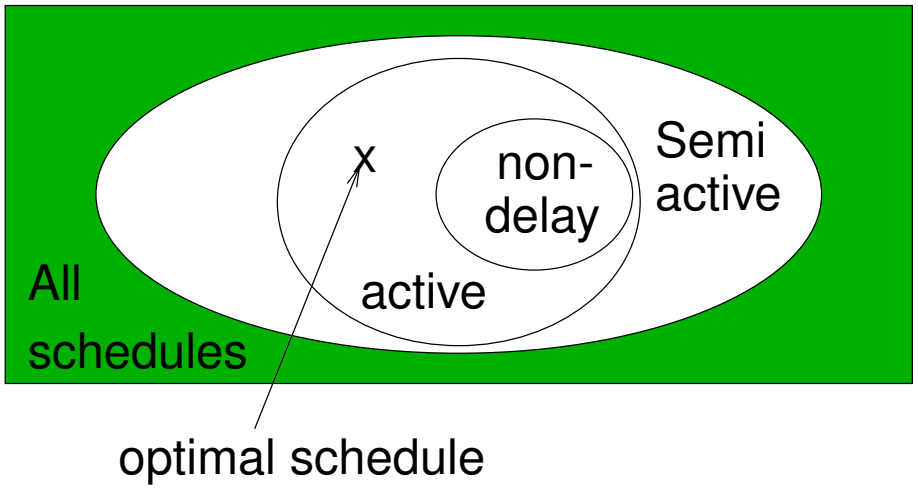


Classes of Schedules

Properties:

- every nonpreemptive nondelay schedule is active
- every active schedule is semiactive
- if the objective criterion is regular, the set of active schedules contains an optimal schedule (regular = non decreasing with respect to the completion times)

Summary:



Research topics for Scheduling

- determine border line between polynomially solvable and NP-hard models
- for polynomially solvable models
  - find the most efficient solution method (low complexity)
- for NP-hard models
  - develop enumerative methods (DP, branch and bound, branch and cut, ...)
  - develop heuristic approaches (priority based, local search, ...)
  - consider approximation methods (with quality guarantee)