Announcements:

please register by sending an email with subject "ADM III" to me (rolf.moehring@tu-berlin.de)

I Introduction to Deterministic and Stochastic Scheduling Problems

§1 Projects and Partial Orders

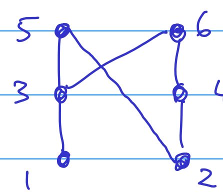
$i < j \;\hat{=}\;$ j must wait for i's completion

└ transitive, asymmetric $\Rightarrow$ partial order on job set V

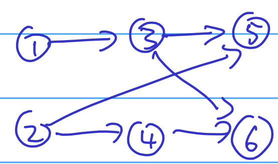
use transitive reduction for representation

in Mathematics

Hasse diagram



computer science, OR, dags



so $1<3$, $1<5$ etc

§2 The deterministic project scheduling model

Every job $j \in V$ has a fixed processing time $x_j \in \mathbb{R}_>$

$\rightarrow$ processing time vector $x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n_>$

Remark: usual notation p_j for processing time
we use x_j because processing times will be variables that may change

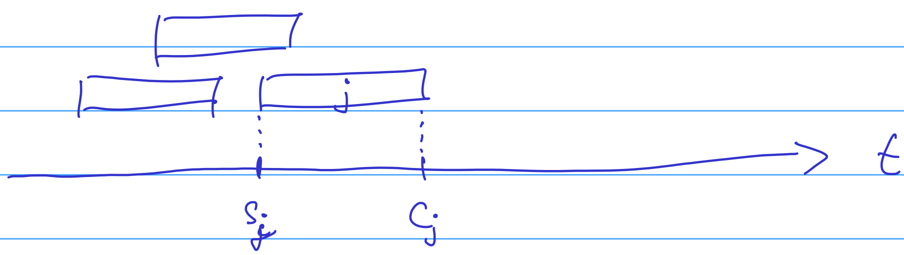
no preemption: no interruption of jobs
so jobs are the smallest units of a project

precedence constraints given by a partial order $\underbrace{G \text{ or } <}_{\text{used synonymously}}$

schedule S = vector $S = (S_1, S_2, \ldots, S_n)$ of start times for the jobs $S_i \geq 0$

S respects G iff $[\, i < j \Rightarrow S_i + x_i \leq S_j \,]$
"j must wait for i"

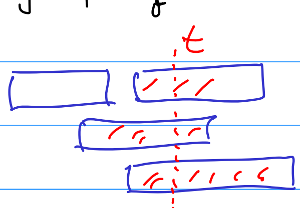
$C_j := S_j + x_j$ is the completion time of job j
representation of schedules by Gantt chart



resource constraints are modeled by a system
$\mathcal{F} = \{F_1, F_2, \ldots, F_k\}$ of forbidden sets or bottlenecks
Each F_i is an antichain of G
$\hookrightarrow$ $u \parallel_G v$ for all $u, v \in F_i$

that must not be scheduled simultaneously at any moment during project execution, but every proper subset can

A schedule S respects $\mathcal{F}$ iff, for every F_i, for every t
$\{ j \mid S_j < t < C_j \} \not\supseteq F_i$



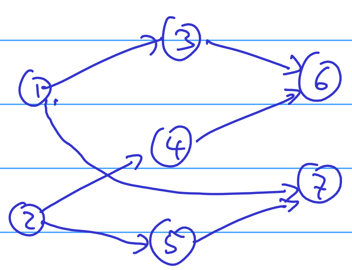
2.1 Lemma: Resource constraints given by forbidden sets model precisely constant resource requirements and availabilities

i.e. constant amount during processing of a job

constant availability during project execution

Proof by example:

Example a)



j	1	2	3	4	5	6	7
$r_1(j)$	2	1	1	–	–	–	–
$r_2(j)$	–	1	1	1	2	2	2

$r_i(j)$ = constant amount of resource i required by job j

availabilities $R_1 = 2$ units of resource 1

$R_2 = 3$ " 2

$\Rightarrow \mathcal{F} = \{ \underbrace{\{1,2\}}_{F_1}, \{3,4,5\}, \{6,7\}, \{3,4,7\}, \underbrace{\{5,6\}}_{F_5} \}$

$\boxed{\text{Every system } \mathcal{F} \text{ (with no } F_i \subseteq F_j \text{) of antichains of } G \text{ can be obtaine in this way, even with } r_i(j) \in \{0,1\}}$

j	1	2	3	4	5	6	7	
$r_1 \hat{=} F_1$	1	1	–	–	–	–	–	$R_1 = 1$
$r_2 \hat{=} F_2$	–	–	1	1	1	–	–	$R_2 = 2$
$r_3 \hat{=} F_3$	–	–	–	–	–	1	1	$R_3 = 1$
$r_4 \hat{=} F_4$	–	–	1	1	–	–	1	$R_4 = 2$
$r_5 = F_5$	–	–	–	–	1	1	–	$R_5 = 1$

$R_i = |F_i| - 1$

b) m parallel identical machines (m-machine problem)

$\hat{=}$

j	1	2	...	n
$r_1(j)$	1	1		1

$R_1 = m$

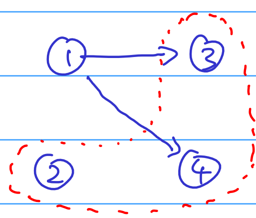
$\Rightarrow$ $\mathcal{F}$ = all $(m+1)$-element antichains of G

Remark: a) $|\mathcal{F}|$ can be exponentiell in n

b) Often it suffices that $\mathcal{F}$ is given implicitly (e.g. by m machines)

Schedule S is <u>feasible</u> for $G, x, \mathcal{F}$ if S respects $G, \mathcal{F}$

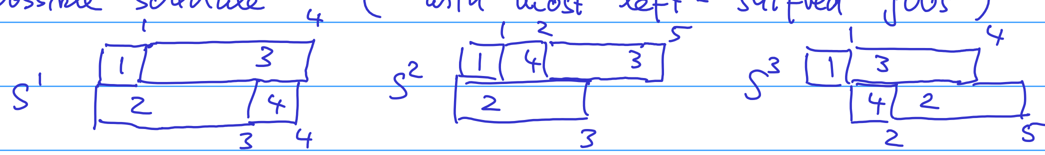
Example:



$\mathcal{F} = \{\{2,3,4\}\}$ $\hat{=}$ 2-machine model

$x = (1, 3, 3, 1)$

possible schedule (with most left-shifted jobs)



all feasible

Differentiate between feasible schedules by a regular measure of performance ("cost" function)

$\kappa : \mathbb{R}^n \to \mathbb{R}^1$ non-decreasing in every component

$\kappa(C_1, \dots, C_n)$ $\hat{=}$ cost of performing the project according to schedule S

$=: \kappa(S, x)$

Expl. (i) $\kappa(C_1, \dots, C_n) = \max\{C_1, \dots, C_n\} =: C_{max}$

makespan, project duration

(ii) $\kappa(C_1, \dots, C_n) = \sum_j C_j$ sum of completion times
(models average completion time)

(iii) $= \sum_j \underbrace{w_j}_{w_j \geq 0} C_j$ weighted sum of completion times

$= \sum_j w_j T_j$ T_j = tardiness of job j
$= \max\{0, C_j - d_j\}$

d_j: due date of job j

Optimization aim:
Find a feasible schedule that minimizes $\kappa(S, x)$

Expl S^1 is optimal for C_{max}
S^2 is optimal for $\sum_j C_j$
S^3 is optimal for $\sum_j w_j C_j$ with the right weights
or also for $\sum_j T_j$ with $d_j = C_j$

$\Downarrow$

optimal schedule depends on objective
may restrict to "left-shifted" schedules since κ is nondecreasing

special case: $\boxed{\text{no resource constraints}}$
Is there a "best" schedule for G, x

Yes: Early start schedule

$$ES_G[x](j) := \begin{cases} 0 & j \text{ is minimal in } G \\ \max\limits_{(i,j) \in E} \{ ES_G[x](i) + x_i \} & \text{otherwise} \end{cases}$$

2.2. Lemma

a) $ES_G[x]$ is a schedule that respects G

b) $\underbrace{ES_G[x]}_{\text{vector}} \leq \underbrace{S}_{\text{vector}}$ for every schedule S respecting G

componentwise

c) $ES_G[x](j)$ = length of a longest chain in $G \mid Pred(j)$

induced subgraph / order induced by $Pred(j)$

set C of jobs with $u \sim_G v$ for $u, v \in C$

$$= \max \{ \sum_{i \in C} x_i \mid C \text{ is a maximal chain in } G|Pred(j) \}$$

w.r.t. $\subseteq$

d) $ES_G[\cdot] : \mathbb{R}^n_{\geq} \to \mathbb{R}^n_{\geq}$ is positive homogeneous, convex, monotone, sublinear, continuous

Proof c) Induction on $|Pred(j)|$

$|Pred(j)| = 0 \Rightarrow Pred(j) = \emptyset \Rightarrow$ max over empty set

$\Rightarrow \max \{ \dots \} = 0$

$\Rightarrow$ j in minimal in G, so by def $ES_G[x](j) = 0$ ✓

inductive step: $|Pred(j)| > 0$

$$ES(j) \underset{\text{Def}}{=} \max_{(i,j) \in E} [ES(i) + x_i]$$

$|Pred(i)| < |Pred(j)| \Rightarrow$ can use inductive assumption on i

Ind. Ass

$$= \max_{(i,j)\in E} \left[\left[\max_{\substack{\uparrow \\ C \text{ max chain in Pred}(i)}} \sum_{k\in C} x_k \right] + x_i \right]$$

every chain ending in j is a chain in Pred(i)
+ arc (i,j) for some $i \in$ InPred (j)

$$= \max_{C \text{ max chain in Pred}(j)} \sum_{k\in C} x_k$$