§ 2 The deterministic project scheduling model

Every job $i \in V$ has a fixed processing time $x_i > 0$

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

[usual notation: p_i for processing time

we use x_i because it will be a variable that may change]

no preemption: jobs are the smallest units

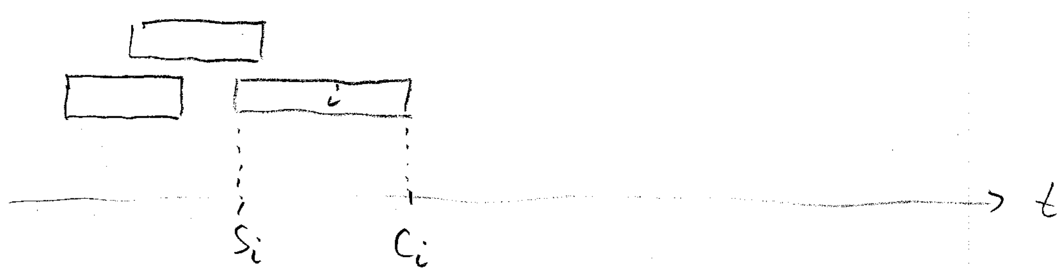
precedence constraints given by partial order $\underbrace{G \text{ or } <}_{\text{used synonymously}}$

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 $[\, i < j \Rightarrow \underbrace{S_i + x_i \leq S_j}_{j \text{ must wait for } i} \,]$

$C_i := S_i + x_i$ is the completion time of job i

representation of schedules by Gantt-charts



resource constraints are modeled by a system $\mathcal{F} = \{F_1, \dots, F_k\}$ of <u>forbidden sets</u> or <u>bottlenecks</u>

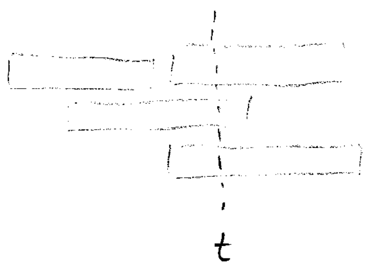
Each F_i is an <u>antichain</u> of G

↳ $u \parallel_G v$ for all $u, v \in F_i$

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

A schedule S <u>respects $\mathcal{F}$</u> iff, for every F_i, for every t

$$\{ j \mid S_j < t < C_j \} \not\supseteq F_i$$



set of jobs j that are processed at time t

2.1 LEMMA: Resource constraints given by forbidden sets model precisely constant <u>resource requirements</u> and <u>availabilities</u>

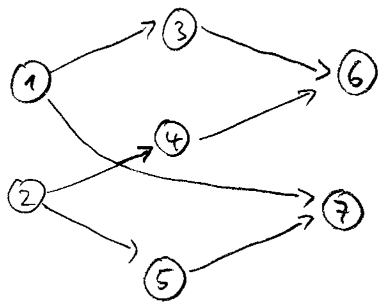
i.e. constant amount during processing of a job

constant availability during project execution

Proof by example:

Example: a)

G:



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

two resource types
constant resource requirements

availability: $R_1 = 2$ units of resource 1
$R_2 = 3$ units of resource 2

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

$\boxed{\text{Every system } \mathcal{F} \text{ (s.t. no } F_i \subseteq F_j \text{) can be obtained in this way, even with } r_i(j) \in \{0,1\}}$

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			1	$R_3 = 2$
$r_4 \hat{=} F_4$					1	1		$R_4 = 1$
$r_5 \hat{=} F_5$						1	1	$R_5 = 1$

$R_i = |F_i| - 1$

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

$\hat{=}$

j	1	2	3	....	n
$r_1(j)$	1	1	...		1

$R_1 = m$

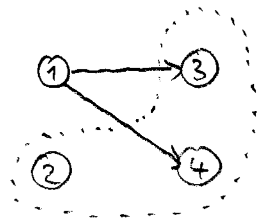
$\Rightarrow \mathcal{F}$ = all $(m+1)$-element antichains of G

<u>Remark</u>: a) $|\mathcal{F}|$ may be exponentiell in n

b) Sometimes 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 and $\mathcal{F}$

Expl:

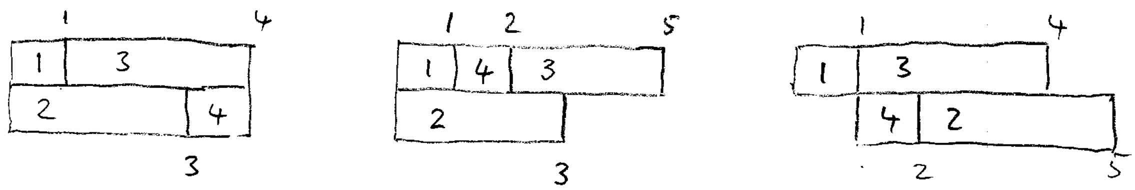


$\mathcal{F} = \{ \{2,3,4\} \}$

$\hat{=}$ 2 machine problem

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

possible schedules (with most left-shifted jobs)



S^1 S^2 S^3

Differentiate between feasible schedules by

<u>regular measure of performance</u>

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

$\kappa(C_1, C_2, \dots, C_n) \hat{=}$ <u>cost</u> of performing the project according to schedule S

$=: \kappa(S, x)$

Expls:

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

project duration, makespan of S

(ii) $\kappa(C_1, \ldots, C_n) = \sum_j C_j$ sum of completion times

$= \sum_j w_j C_j$ weighted sum of completion times

(iii) $= \sum_j w_j T_j$ T_j = tardiness $= \max\{0, C_j - d_j\}$

d_j: due date for 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 $\sum C_j$

S^3 is optimal 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 regular

Special case: $\boxed{\text{no resource constraints}}$

Is there a "best" schedule for G, x ?

YES: <u>Early start schedule</u>

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

2.2 LEMMA

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

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

c) $ES_G[x](j)$ = length of a longest chain in $\underbrace{G|\mathrm{Pred}(j)}_{\substack{\text{subgraph / order} \\ \text{induced by } \mathrm{Pred}(j)}}$

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

$\subseteq$-wise, $u \sim_G v$ for all $u, v \in C$

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

Proof: c) Induction on $|\mathrm{Pred}(j)|$

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

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

$\Rightarrow j$ minimal in $G \overset{\text{Def}}{\Rightarrow} ES(j) = 0$ ✓

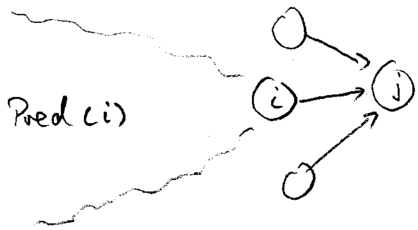
$|\mathrm{Pred}(j)| > 0$

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

$\uparrow \ |\mathrm{Pred}(i)| < |\mathrm{Pred}(j)|$

ind. hyp.

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



every chain ending in j is a chain in Pred(i)

+ arc (i,j) for some $\underbrace{i \in \text{ImPred}(j)}_{\iff (i,j) \in E}$

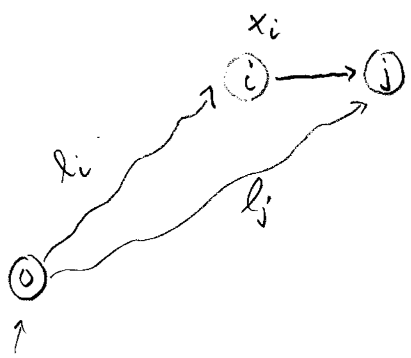
$$= \max_{C \text{ max chain in Pred}(j)} \sum_{k\in C} x_k$$

b) follows from c) since length of a longest chain in Pred(j) is a lower bound on S_j for every schedule S

a) Show that ES respects G

follows from recursive definition

or, alternatively, from c)



assume dummy source job (w.l.o.g.)

l_i = length of longest chain in Pred(i)

l_j = ...

$\Rightarrow l_j \geq l_i + x_i$

$\overset{c)}{\Rightarrow}$ $ES(j) \geq ES(i) + x_i$

d) $ES_G[x](j) = \underbrace{\max_C \underbrace{\sum_{i \in C} x_i}_{\text{linear function of } x}}_{\text{max of linear functions}}$

$\Rightarrow$ all properties hold $\blacksquare$

CONSEQUENCES:

a) Given κ, the minimum cost of planning according to $ES_G[x]$ is

$$\kappa(ES_G[x], x) =: \kappa^{ES_G}(x) \quad \text{or} \quad \kappa^G(x)$$

(cost by planning according to G)

b) Can consider $\kappa^G(\cdot) : \mathbb{R}^n \to \mathbb{R}^1$ as a cost or performance function

c) Special case $\kappa = C_{max} \Rightarrow$

$C_{max}^G(x)$ = length of a longest chain in G

CPM method

Resume:

If the world is simple (no resource constraints) then the early bird rule is optimal

Homework:

2.1 S minimizes L_{max} $\Rightarrow$ S minimizes T_{max}

Lateness $L_j := C_j - d_j$

2.2 $\sum w_j C_j$ and $\sum w_j L_j$ are equivalent (same w_j)

2.3* Describe the makespan polytope

$$\{ x \in \mathbb{R}^n_{\geq} \mid C_{max}^G (x) \leq t \} \quad t \text{ fixed } (t=1)$$

by its vertices

Hint: use antichains of G