§6 PRIORITY POLICIES

A planning rule Π is a <u>priority planning rule</u> or <u>priority rule</u> for $[G, F]$ if

(1) Π is elementary [decision times are $t=0$ and job completions]

(2) At every decision time t, there is a priority list $L(t)$

$$L(t): j_1 < j_2 < \dots < j_k$$

on the set of still unscheduled jobs [= not yet started]

(3) Π considers jobs j in $L(t)$ one by one in the order of L and sets $S_j := t$ if possible w.r.t. $[G, F]$ and the previously started jobs

Π is <u>static</u> if every $L(t)$ is a sublist of $L(0)$. Otherwise Π is called <u>dynamic</u>

REMARK: For priority policies [i.e. non-anticipative], the list $L(t)$ may only depend on the history up to time t

Expl: Smith's rule is a priority rule, but not a priority policy

Smith's rule based on expected processing times $E(X_j)$,

i.e. $$\frac{E(X_{i_1})}{w_{i_1}} \le \dots \le \frac{E(X_{i_n})}{w_{i_n}}$$

is a priority policy

<u>Advantages of priority policies</u>

(1) **Every priority policy is minimal among all policies**

Suppose $\pi' \leq \pi$, π priority policy. Consider fixed x

at $t=0$ $\pi[x]$ starts as many jobs as possible at $t=0$

$\pi' \leq \pi \Rightarrow$ $\pi'[x]$ has the same start set

Look at first completion w.r.t. x (π' cannot start new job earlier, since resources are all used)

$\Rightarrow$ same argument: $\pi[x]$, $\pi'[x]$ start same set of jobs at first completion

$\Rightarrow$ (iteration) $\pi'[x] = \pi[x]$ $\overset{\text{all } x}{\Rightarrow}$ $\pi' = \pi$

(2) **easy to implement**

(3) **can give performance bounds for some problems**

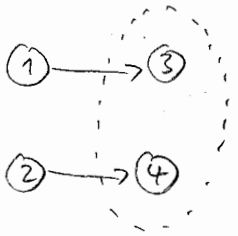
e.g. $\dfrac{C^{\pi}_{max}(x)}{OPT_{C_{max}}(x)} \leq 2 - \dfrac{1}{m}$ for m-machine problems

[proof later]

Disadvantages of priority policies

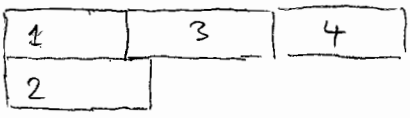
(1) **priority policies are neither continuous nor monotone and thus may cause instabilities**

Expl:

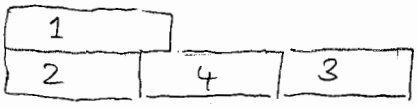


← only forbidden set $L: 1 < 2 < 3 < 4$

$x_1 < x_2 \Rightarrow 0$



$x_2 < x_1 \Rightarrow 0$

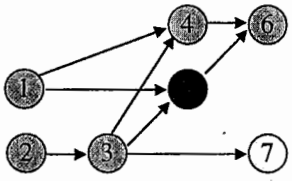


$\Rightarrow$ for fixed x_2, x_3, x_4 and $x_1 \in [x_2 - \varepsilon, x_2 + \varepsilon]$ is $\Pi[\cdot](4)$ neither continuous nor monotone

(2) Graham anomalies

(a)

Priority policies are neither continuous nor monotone (Graham anomalies)

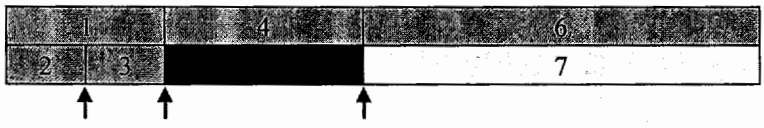


$\mathcal{F}$: 2 identical machines

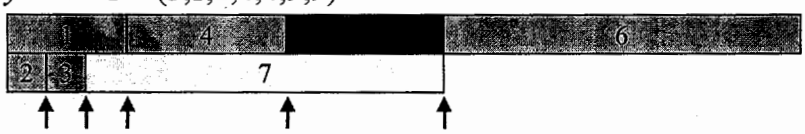
min C_{max}

$L = 1 < 2 < ... < 7$

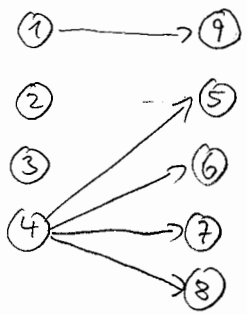
$x = (4,2,2,5,5,10,10)$



$y = x - 1 = (3,1,1,4,4,9,9)$



(b)



$x = (3, 2, 2, 2, 4, 4, 4, 4, 9)$

$m = 2$

$k = C_{max}$

$L = 1 < 2 < \ldots < 9$

$\Rightarrow C_{max}(x)$ grows when $4 < 5$ and $4 < 6$ are deleted

(c) Take example (b) but with $m = 3$

$\Rightarrow C_{max}(x)$ grows when $m = 4$ machines are available

(3) In general, there is no priority policy that yields an optimal schedule for fixed x

Take Example (a) with processing times y

Exercises

6.1 Show that there are no Graham anomalies for m-machine problems without precedence constraints.

Show that, in this case, every priority policy is continuous and monotone