Energy Efficient Online Deadline Scheduling*

Prudence Wong

CTAG Seminar (1 March 2007)

Joint work with Ho-Leung Chan, Tak-Wah Lam, Lap-Kei Lee, Fred Mak (University of Hong Kong) and Joseph Wun-Tat Chan (King's College London)

* SODA 2007

Online deadline scheduling revisited

- Each job J arrives at unpredictable time with its own requirement: **work** and **deadline**. ← w(J) & d(J)
- **Aim:** Schedule the jobs on a single processor to meet their deadlines; preemption is allowed.
  - Underloaded systems: If there exists a schedule completing all jobs by the deadlines, then EDF (earliest deadline first) can always do so.
- **Overloaded systems:** too many jobs; mission impossible (even using an offline algorithm).
  - Objective: maximum throughput, i.e., total work of jobs completed by their deadlines.

Competitiveness

- An online algorithm A is c-competitive if $\exists$ constant b, for any job sequence I,

$$A(I) \geq 1/c \; Opt(I) + b,$$

where $A(I)$ and $Opt(I)$ are the throughput of A and the optimal offline algorithm.

- D^{over} is 4-competitive and is optimal [Koren & Shasha SICOMP95].

Energy efficiency

For mobile devices, energy efficiency is a major concern.

How to save energy?

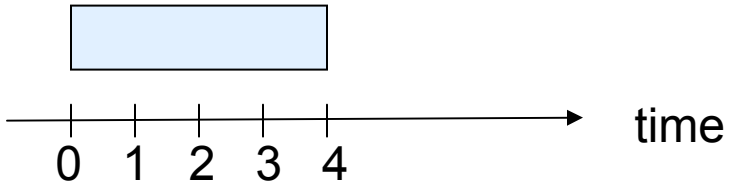
Dynamic speed (voltage) scaling.

- Slow down the processor whenever possible.
- To run at speed s, rate of energy usage power = s^{α} where $\alpha > 1$ (literature: $\alpha = 2$ or 3).

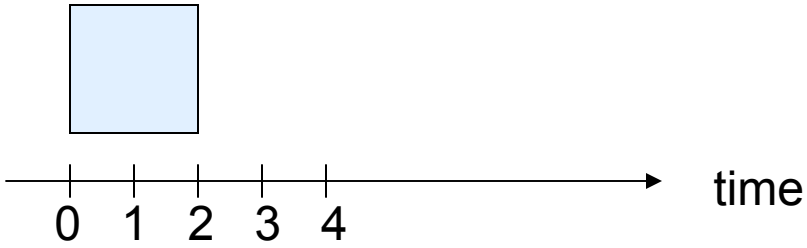
Example

Assume $\alpha = 3$.

Consider a job with 4 units of work.



Energy usage: $1^3 \times 4 = 4$



Energy usage: $2^3 \times 2 = 16$

Objectives

- Assume that the processor can vary the speed within $[0,T]$ for some $T \geq 1$.
- Optimal schedule (Opt):
  - Objective 1: **maximum throughput**
  - Objective 2: use the **minimum energy** to achieve the maximum throughput.
- **Problem :** Devise an online algorithm that is **O(1)-competitive** with respect to both **throughput** and **energy**.

Previous work

Unbounded maximum speed, i.e., $T = \infty$.

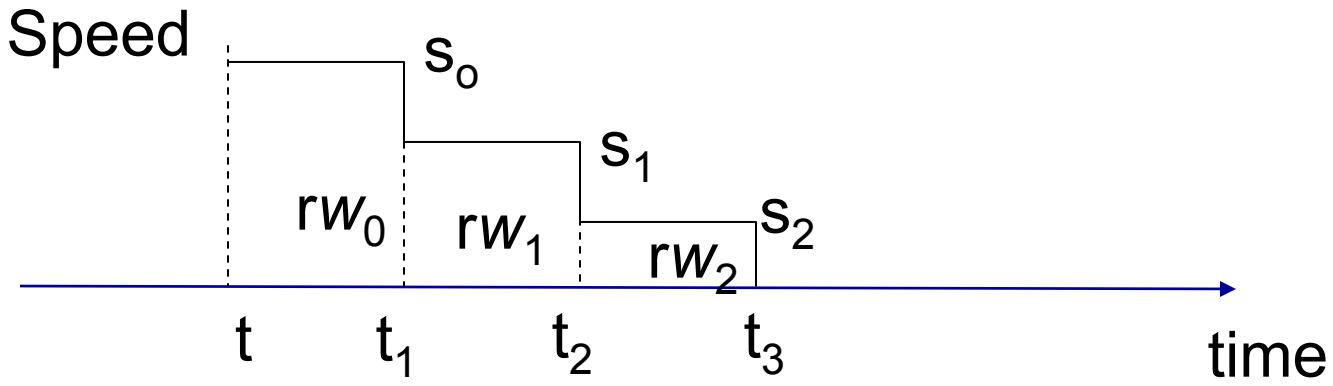
- completing all jobs is always feasible.
- The only concern is how to minimize energy.

O(1)-competitive algorithms w.r.t. energy:

- [Yao et al. FOCS95] AVR is $2^{\alpha}\alpha^{\alpha}$-competitive.
- [Yao et al. 95, Bansal et al. 04] OA is α^{α}-competitive.
- [Bansal et al. FOCS04] $2(\alpha/(\alpha-1))^{\alpha}e^{\alpha}$-competitive; lower bound: $\Omega((4/3)^{\alpha})$

NB. α is the constant such that power(s) = s^{α}.

OA: EDF + speed function

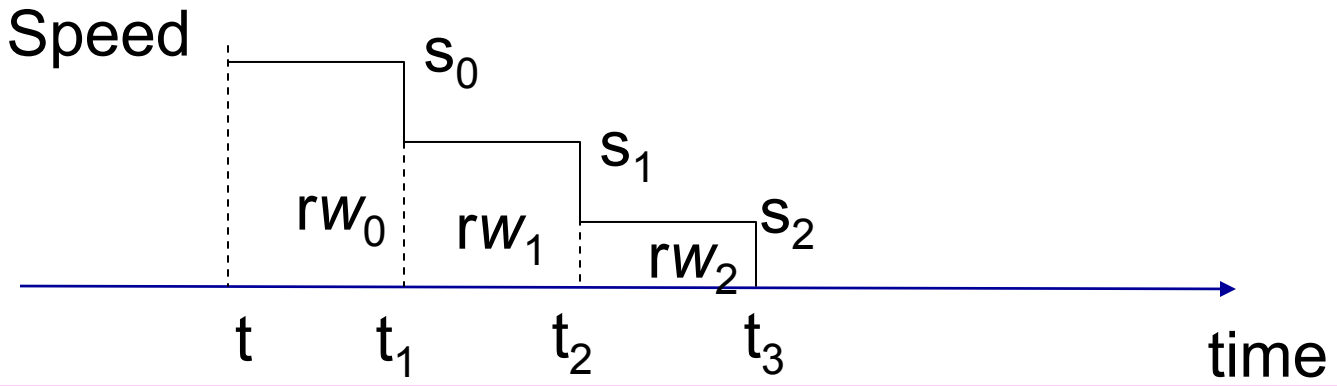


At any time t, the speed is determined as follows:

- rw(t,t') = the remaining work with deadline in (t,t'].
- density(t,t') = rw(t,t') / (t'-t).

Minimum average speed to avoid missing a deadline in (t,t'].

OA: highest density first



At any time t, the speed is determined as follows:

- rw(t,t') = the remaining work with deadline in (t,t'].
- density(t,t') = rw(t,t') / (t'-t).
- t_1 = the first t' > t such that density(t,t') is maximum
- During (t, t_1), speed = density(t, t_1).
- Similarly, t_2 = the first t' > t_1 such that density(t_1,t') is maximum, and during (t_1, t_2), speed = density(t_1, t_2); ...

Our results

- **Finite** maximum speed (i.e., [0,T]).
  [e.g. Pillari & Shin SOSP01]
- In this case, the system may be overloaded
  - even the optimal scheduler Opt cannot complete all jobs by the deadlines.
- **Our result:** $O(1)$-competitive on throughput & energy usage.
  - The algorithm is called **FSA(OAT)**.
  - **14**-competitive on throughput, and
  - $(\alpha^\alpha + \alpha^2 4^\alpha)$-competitive on energy.

Extensions

Intel XScale, Intel SpeedStep, AMD Athlon 64, Transmeta LongRun2 and Efficeon Processor

- Discrete speed levels: $s_1, s_2, \ldots, s_d$ [Li & Yao 05]
  - **14**-competitive on throughput, and
  - $(\Delta^{\alpha}(\alpha^{\alpha} + \alpha^2 4^{\alpha}) + 1)$-competitive on energy where $\Delta = \max_{i>1}(s_{i+1} / s_i)$
- Better throughput via resource augmentation
  - online scheduler: max speed is relaxed to **$(1+\varepsilon)T$**
  - **$(1+1/\varepsilon)$**-competitive on throughput
  - **$(1+\varepsilon)^{\alpha}(\alpha^{\alpha} + \alpha^2 4^{\alpha})$**-competitive on energy

Further extension

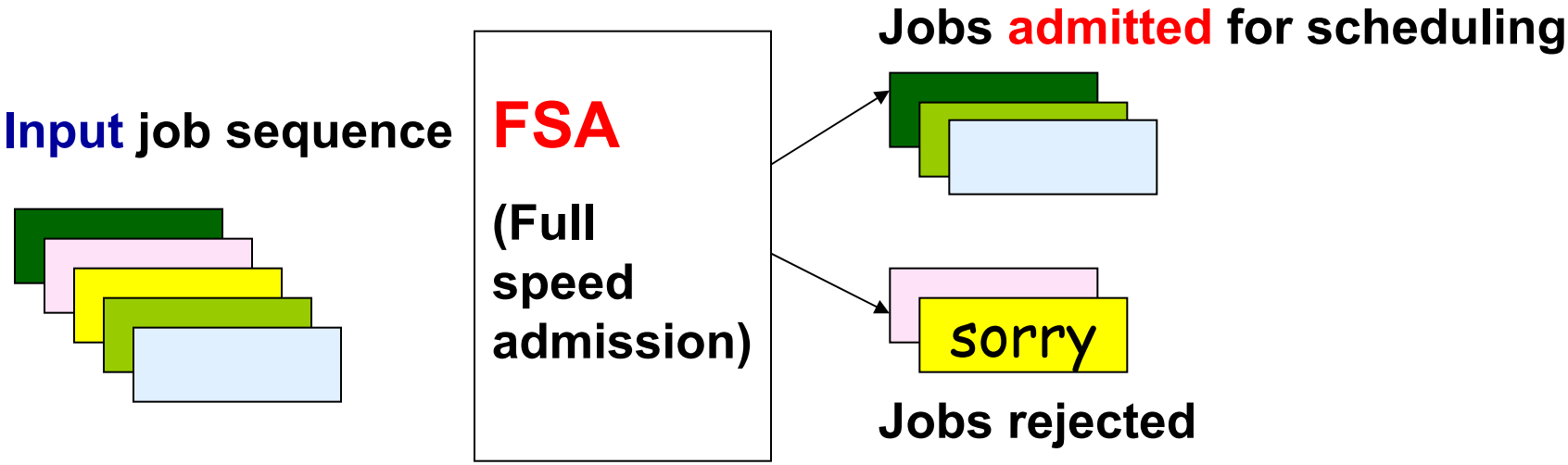
- Jobs with arbitrary values; throughput is measured by the total value of jobs completed.
  - 14*k*-competitive on throughput.
  - $(\alpha^\alpha + \alpha^2 2^\alpha (1+k)^\alpha)$-competitive on energy,

    where *k* is the importance ratio,

    i.e., the ratio of the largest to the smallest possible value density.

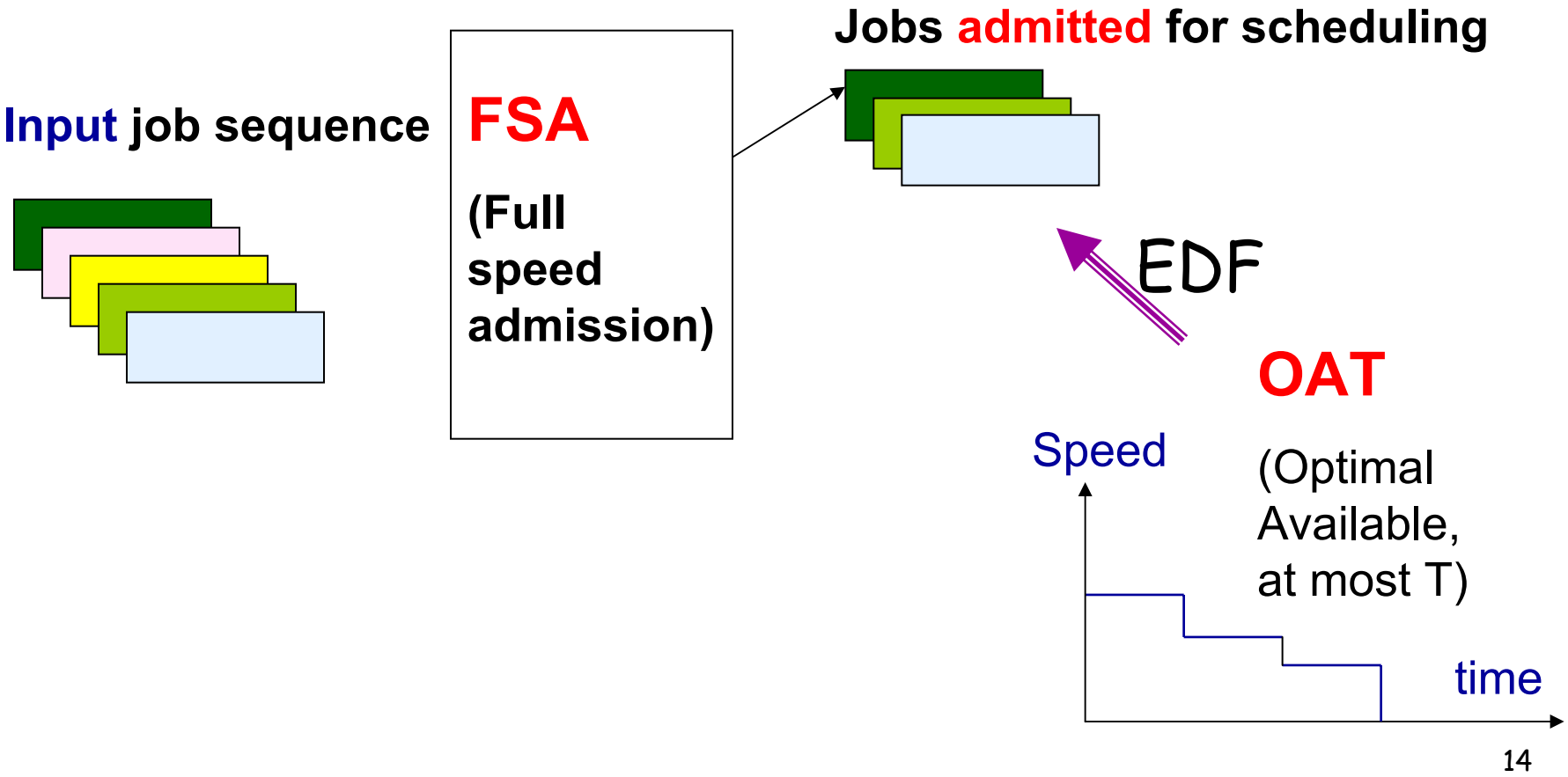
The algorithm FSA(OAT)

Job selection strategy + Speed function

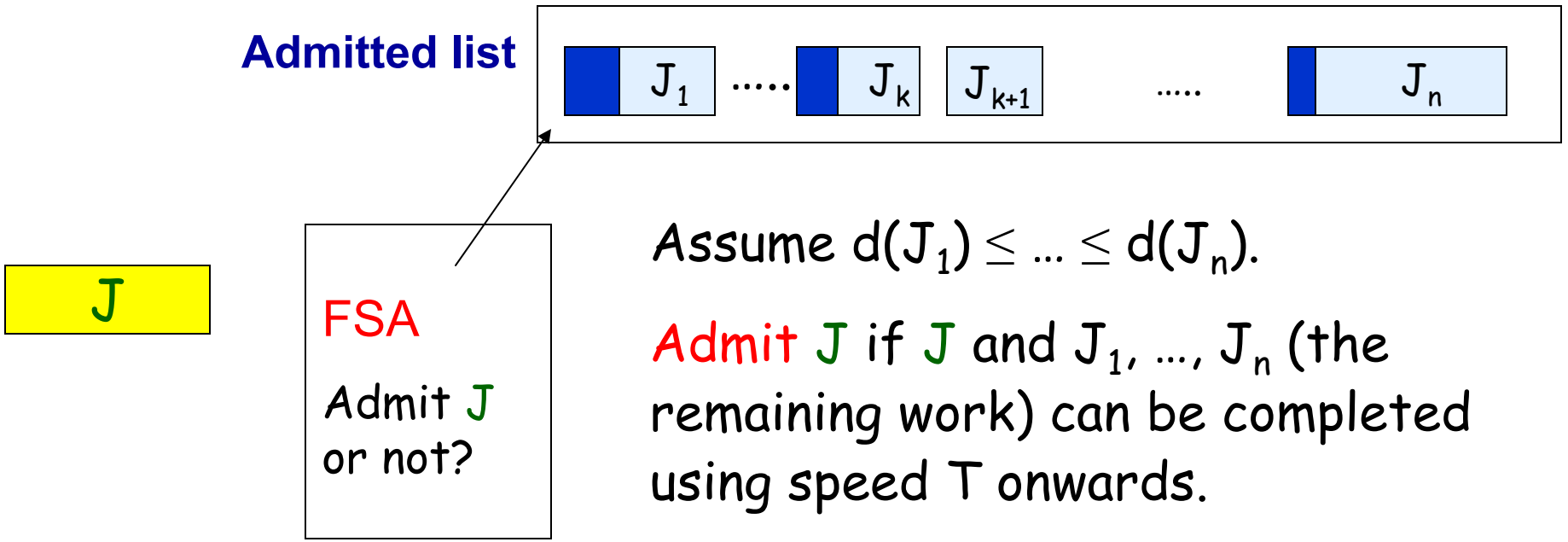


The algorithm FSA(OAT)

Job selection strategy + Speed function



FSA - Full speed admission



Assume $d(J_1) \leq \ldots \leq d(J_n)$.

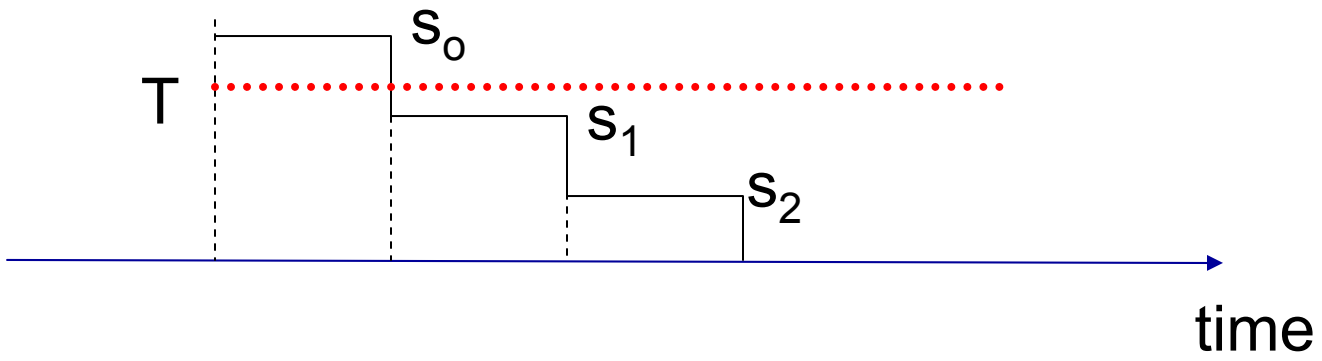
Admit J if J and $J_1, \ldots, J_n$ (the remaining work) can be completed using speed T onwards.

If $w(J) > 2(w(J_1) + \ldots + w(J_k))$ and $J, J_{k+1}, \ldots, J_n$ can be completed using speed T, then admit J and expel $J_1 \ldots J_k$.

PS. $w(J) \Rightarrow$ value(J)

OAT

- At any time t, the speed of OAT is the minimum of the speed of OA and T.



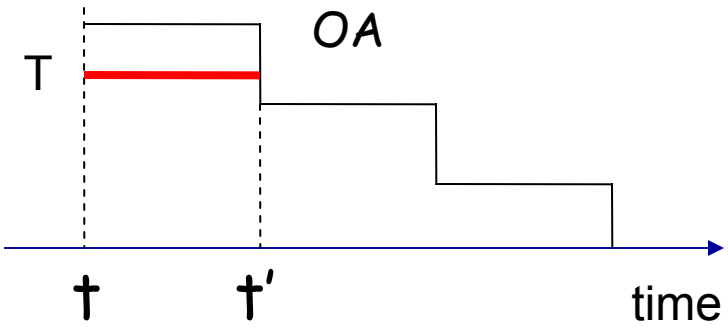
- Unlike OA, OAT doesn't intend to complete all jobs.
- Note that OAT doesn't depend how FSA admits jobs.

Analysis of FSA(OAT)

1. The speed function **OAT** is fast enough to complete all jobs admitted (but not expelled) by **FSA**. We say that FSA(OAT) is **honest**.
2. For any speed function **f**, if FSA(**f**) is **honest**, the total work admitted
   $\geq$ **1/14** x the total work completed by Opt.
3. Energy usage of OAT is at most $(\alpha^{\alpha} + \alpha^2 4^{\alpha})$ times that of Opt.

OAT makes FSA honest

- By induction on job arrival time. Consider any such time t. Suppose OAT is at speed T during $[t,t']$.



Two invariants of the admitted list S_t:

- For jobs in S_t with deadline $\leq t'$, total remaining work $\leq T(t' - t)$ and OAT can complete them.
- For each job J in S_t with deadline $> t'$, OA is too busy to work on J during $[t,t']$ and OA can't outperform FSA(OAT) on J using extra speed beyond T.

- OA can complete all jobs, and OAT can complete S_t.

Competitiveness w.r.t. throughput

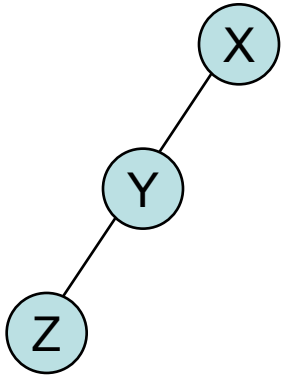
Given a job sequence I, FSA(OAT) divides I into

C : the jobs admitted and never expelled;

E : those admitted but expelled eventually; and

N : those not admitted.

- Fact: (i) FSA(OAT) completes C only.
  (ii) $w(E) \leq w(C)$.



X expels Y
Y expels Z

Competitive ratio = 14

Consider the span of the jobs in N. Assume their union has a total length l.

- Trivial: $w(Opt) \leq Tl + w(C) + w(E)$.
- Non-trivial: $Tl \leq 6(w(C) + w(E))$.

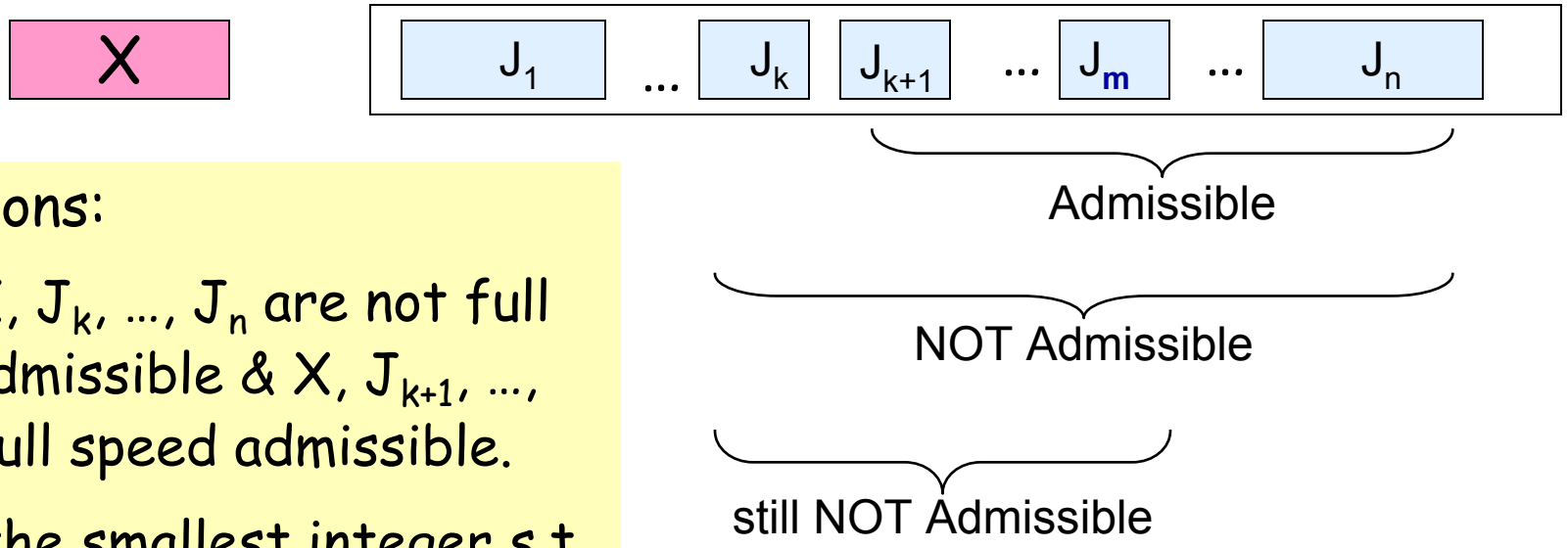
Conclusion: $w(Opt) \leq 7(w(C) + w(E)) \leq 14\, w(C)$.

NB. Span(J) is the interval $\rho(J) = [r(J), d(J)]$.

C : the jobs admitted and never expelled;
E : those admitted but expelled eventually; and
N : those not admitted.

$T\mathcal{I} \leq 6w(A)$, where $A = C \cup E$

Consider any job X in N. Note that X is **not** admitted.



Definitions:

k $\leq$ n : X, J_k, …, J_n are not full speed admissible & X, J_{k+1}, …, J_n are full speed admissible.

m $\leq$ n : the smallest integer s.t. X, J_k, …, J_m are not full speed admissible.

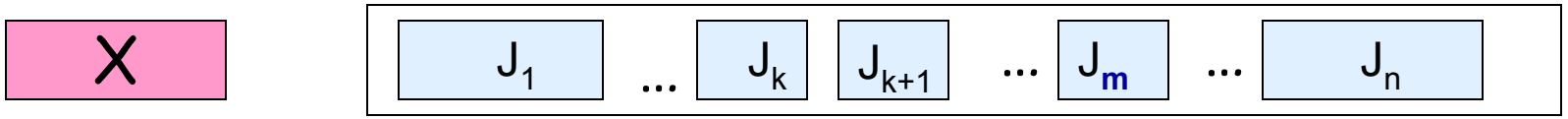
ρ**(X)** = [r(X), d(X)]

ρ'**(X)** = [r(X), max{d(X), $d(J_m)$}]

N.B. $d(J_i)$ in $\rho'(X)$ for $i \leq m$

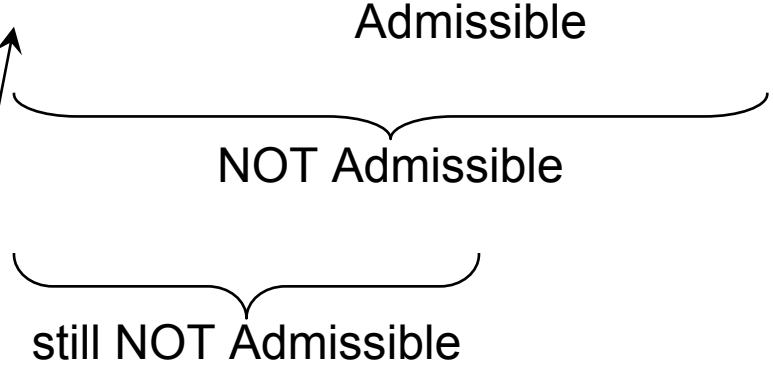
$T\mathcal{I} \leq 6w(A)$, where $A = C \cup E$

Consider any job X in N. Note that X is **not** admitted.



Lemma.

$T|\rho'(X)| \leq 3\, w(\, A|\rho'(X)\,)$
where $A|\rho'(X)$ are the jobs in A with deadline in $\rho'(X)$.



$$T|\rho'(X)| \leq w(X) + w(J_k) + w(J_{k+1}) + \ldots + w(J_m)$$

$$\leq 2[\, w(J_1) + \ldots + w(J_k)\,] + (w(J_k) + \ldots + w(J_m))$$

$$\leq 3w(\, A|\rho'(X)\,)\,.$$

$\rho(X) = [r(X), d(X)]$

$\rho'(X) = [r(X), \max\{d(X), d(J_m)\}]$

N.B. $d(J_i)$ in $\rho'(X)$ for $i \leq m$

$T\ell \leq 6w(A)$

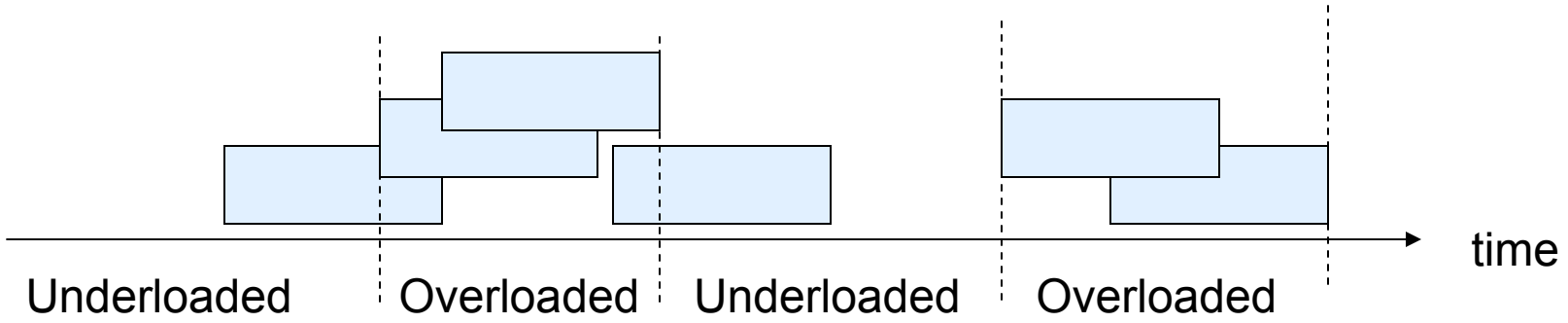
N contains a subset $\{X_1, X_2, \ldots\}$ such that $\rho'(X_1)$, $\rho'(X_2)$, … do not overlap and $|\rho'(X_1)| + |\rho'(X_2)| + \ldots \geq \frac{1}{2}\ell$.

$$
\begin{aligned}
T\ell &\leq 2\,[\, T|\rho'(X_1)| + T|\rho'(X_2)| + \ldots \,] \\
&\leq 2\,[\, 3w(A|\rho'(X_1)) + 3w(A|\rho'(X_2)) + \ldots \,] \\
&\leq 6\, w(A).
\end{aligned}
$$

Energy usage

- OA: α^α -competitive (against any algorithm that completes all jobs with unbounded max speed).
- OAT: $(\alpha^\alpha + \alpha^2 4^\alpha)$-competitive (against any algorithm that maximizes the throughput with max speed T).
- The proof is based on a better understanding of Opt in the so-called "**overloaded**" and "**underloaded**" periods and an adaptation of the analysis in [Bansal et al. 04].

Energy usage of OAT



Roughly speaking …

- Overloaded periods: **Opt** completes at least $\frac{1}{4}$ LT units of work, where L is the total length of the overloaded periods.
- **Opt** completes all jobs in underloaded periods. The analysis of OAT is similar to OA.

Future work

- Better upper/lower bound on throughput (while retaining O(1)-competitive w.r.t. energy) [Bansal, Chan, Lam]
  - 4-competitive? Lower bound is 4 (without energy concern).
  - underloaded system (where Opt can complete all jobs), $(1+\varepsilon)$-competitive w.r.t. throughput? 1-competitive is not possible.
- Energy efficiency [Chan, Lam, Mak, Wong]
  - Put energy efficiency primary concern
  - Maximize the throughput given a certain constraint on energy efficiency.

Future work

- Sleep state
  - [Irani et al. 03]: $T = \infty$; $O(1)$-competitive w.r.t. energy
- Flow time + finite max speed or discrete speed levels
- Temperature concern