Energy Efficient
Online Deadline Scheduling*

Prudence Wong
University of Liverpool

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 there exists a constant $b$, for any job sequence $I$,

$$A(I) \geq \frac{1}{c} \text{Opt}(I) + b,$$

where $A(I)$ and $\text{Opt}(I)$ 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:
    \[ \text{power} = s^\alpha \text{ where } \alpha > 1 \]
    (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 $2^{\alpha}\alpha$-competitive.
- [Yao et al. 95, Bansal et al. 04] OA is $\alpha$-competitive.
- [Bansal et al. FOCS04] $2^{(\alpha/(\alpha-1))\alpha e^{\alpha}}$-competitive; lower bound: $\Omega((4/3)^{\alpha})$

NB. $\alpha$ is the constant such that $\text{power}(s) = s^\alpha$. 
At any time $t$, the speed is determined as follows:

- $\text{rw}(t,t') = \text{the remaining work with deadline in } (t,t']$.
- $\text{density}(t,t') = \text{rw}(t,t') / (t'-t)$.

Minimum average speed to avoid missing a deadline in $(t,t']$. 
At any time $t$, the speed is determined as follows:

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

- Discrete speed levels: $s_1, s_2, ..., s_d$ [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

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

- Jobs with arbitrary values; throughput is measured by the total value of jobs completed.
  - $14k$-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

Input job sequence

FSA

(Job Full speed admission)

Jobs admitted for scheduling

Jobs rejected

Prudence Wong @ Bristol Algorithm Days 2007
The algorithm FSA(OAT)

Input job sequence

FSA
(Full speed admission)

Jobs admitted for scheduling

EDF

OAT
(Optimal Available, at most T)

Speed

time

Prudence Wong @ Bristol Algorithm Days 2007
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.  \( w(J) \Rightarrow \text{value}(J) \)
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 on 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
\[ \geq \frac{1}{14} \times \text{the total work completed by Opt.} \]

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 $I$, $\text{FSA(OAT)}$ divides $I$ into

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

- **Fact:**
  1. $\text{FSA(OAT)}$ completes $C$ only.
  2. $w(E) \leq w(C)$.

![Diagram showing the relationships between X, Y, and Z with X expels Y and Y expels Z]
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.
Energy usage

- **OA**: $\alpha^\alpha$-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].
Roughly speaking ...

- **Overloaded periods**: $\text{Opt}$ 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 '07]
  - 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