Preemptive Scheduling on Identical Machines Subject to Migration Delay

Aleksei V. Fishkin
Joint work with René Sitters
Klaus Jansen
Sergey V. Sevastianov
Problem Definition

- Identical machines, $m$
- Independent jobs, $p_1, p_2, \ldots, p_n$
- Preemptions
- If job migrates then delay $d$
- Objective: $\min C_{\text{max}}$
Problem Definition

Non-Preemptive

Preemptive

Preemptive with $d$-constraint
Scheduling with Communication Delays

\[ P | \text{prec, } p_j=1, c_{jk} = 1 | C_{\text{max}} \text{ is APX-hard. (Hoogeveen, Lenstra, Veltman [1994])} \]

\[ Pm | \text{tree, } p_j = 1, c_{jkiq} \leq D | C_{\text{max}} \text{ is in P. (Engels et al. [2001])} \]

\[ P2 | \text{tree, } p_j = 1, c_{jk} = d | C_{\text{max}} \text{ is NP-hard. (Afrati et al. [2000])} \]

\[ 1 | \text{chains, } p_j = 1, c_{jk} = d | C_{\text{max}} \text{ ??} \]
If \( d = 0 \) then easy: McNaughton’s rule
\[
C_{\text{max}} = \Delta : = \max\{\Sigma p_j / m, p_{\text{max}}\}.
\]
If $d$ is large then hard: Partition problem
Properties of Optimal Schedule

Claim.
For two machines one preemption (migration) suffices.

Proof.
Idea: Take optimal schedule and modify it.
Properties of Optimal Schedule

Claim.

For two machines one preemption (migration) suffices.

Proof.
Step 1: Place migration jobs at the end. It can be infeasible, but we will correct it.
Properties of Optimal Schedule

Claim.

For two machines one preemption (migration) suffices.

Proof.

Step 2: Reschedule the migration jobs using McNaughton’s rule.
Properties of Optimal Schedule

Claim.

For two machines one preemption (migration) suffices.

Proof.

Step 3: Reschedule to satisfy delay constraint.

\[ C_{\text{max}} - p_j \]
Properties of Optimal Schedule

Claim.

For three machines two migrations suffice.

Proof.

Step1: Take optimal schedule and place migration jobs at the end.
Properties of Optimal Schedule

Claim.

For three machines two migrations suffice.

Proof.

Step 2: Reschedule preempted jobs cleverly, using only two preemptions.
Properties of Optimal Schedule

Claim.
For three machines two migrations suffice.

Proof.
Step 2: Reschedule preempted jobs cleverly, using only two preemptions.

Infeasible;
$d$-constraint not satisfied.
Properties of Optimal Schedule

Claim.

For three machines two migrations suffice.

Proof.
- Order machines by length of available interval: \( L_1 \geq L_2 \geq L_3 \).
- Longest Processing Time first and in order \( L_2, L_1, L_3 \).
Claim.

For three machines two migrations suffice.

Proof.
- Order machines by length of available interval: $L_1 \geq L_2 \geq L_3$.
- Longest Processing Time first and in order $L_2, L_1, L_3$. 
Properties of Optimal Schedule

Claim.

For three machines two migrations suffice.

Proof.

- Order machines by length of available interval: $L_1 \geq L_2 \geq L_3$.
- Longest Processing Time first and in order $L_2, L_1, L_3$.
Properties of Optimal Schedule

Claim.

For **three** machines **two** migrations suffice.

Proof.

- Order machines by length of available interval: \( L_1 \geq L_2 \geq L_3 \).
- Longest Processing Time first and in order \( L_2, L_1, L_3 \).

Feasible;
\( d \)-constraint satisfied.
Claim.

For three machines two migrations suffice.

Proof.
- The pseudo schedule preempts at most twice.
- If a job is preempted twice this also appears in OPT.
Properties of Optimal Schedule

Conjecture.

For $m$ machines, $m-1$ migrations suffice.
Properties of Optimal Schedule

**Conjecture.**

For $m$ machines $m-1$ migrations suffice.

Extending the same proof fails.
Properties of Optimal Schedule

Conjecture.

For $m$ machines $m-1$ migrations suffice.

Proving this conjecture will lead to pseudo poly. algorithms

Extending the same proof fails.

Even if we know how long job $j$ is processed on machine $i$, for any $i$ and $j$, then finding the optimal schedule is still a problem. No simple rules.
Properties of Optimal Schedule

Theorem.

There exists an optimal schedule without idle time.
Theorem.

There exists an optimal schedule without idle time.

Proof.

Minimize in this order:
• makespan
• average completion time
• -- mean busy date
Theorem.

There exists an optimal schedule without idle time.

Proof.

Minimize in this order:

• makespan
• average completion time
• $\text{-- mean busy date}$
Theorem.

There exists an optimal schedule without idle time.

Proof.

Minimize in this order:

• makespan
• average completion time
• -- mean busy date
**Theorem.**

There exists an optimal schedule without idle time.

**Proof.**

Minimize in this order:

- makespan
- average completion time
- -- mean busy date
McNaughton’s rule: $C_{\text{max}} = \Delta := \max\{\sum p_j/m, p_{\text{max}}\}$.

**Theorem.**

- If $d \leq \Delta - p_{\text{max}}$ then McNaughton’s rule gives optimal schedule.

- On the other hand, if we restrict to instances with $d \geq (1+\varepsilon)(\Delta - p_{\text{max}})$, for any $\varepsilon > 0$, then the problem is strongly NP-hard.
A Hardness Threshold

Proof.

\[ d = \Delta / (2(a-1)) \quad \Delta - p_{\text{max}} = \Delta / (2a) \]

\[ d = a/(a-1) (\Delta - p_{\text{max}}) < (1+\epsilon)(\Delta - p_{\text{max}}), \quad \text{for } a > 1/ \epsilon + 1. \]
Approximation Results

**Theorem.**
For $m=2$ we can find a $(1+1/\log n)$-approximation in linear time.

**Theorem.**
For arbitrary $m$ we can find a $(1+\varepsilon)$-approximation in $O(m^{f(\varepsilon)}) + O(n \log m)$ time.