Online Scheduling with Redirection for Parallel Jobs

Adrien Faure, Giorgio Lucarelli, Olivier Richard, Denis Trystram

HIPS 25th, May 21, 2020
Introduction

- High Performance Computing (HPC) cluster

Users submit jobs → Job Scheduler

Schedules and launch jobs

Job queue: [ ]

Executes jobs

HPC Production Platform

- Need for efficient HPC scheduling policies
- Online job scheduling with job redirection
Introduction

- Novel scheduling algorithm: parallel jobs with redirection
- No preemption: job restarted on dedicated resources pool
HPC scheduling

**Online scheduling** → jobs unknown until submission

**Job definition:**

![Diagram showing job execution timeline with labels for start, wall time, and resources.

First Come First Serve (FCFS)

- Not enough resources → job waits
- Jobs ordered by arrival time
First Come First Serve
First Come First Serve

Job queue:

![Graph showing job scheduling]

Job queue:
First Come First Serve

- At time $t_0$:
  - The job queue is empty.
  - A new job arrives.

- At time $t_1$:
  - A new job arrives.
  - The job from $t_0$ is still in progress.

- At time $t_2$:
  - The job from $t_1$ is now in progress.
  - The job from $t_0$ has completed.
First Come First Serve

Job queue:

Job queue:

Job queue:
Redirection

- **Parallel** & independent jobs
- **Online** scheduling mechanism
  - Compatible with any scheduling algorithm
  - After normal scheduling phase → redirect if needed
- No preemption: job restarted on dedicated resources pool
Redirection

- **Running job** holds a counter $\rightarrow$ increased on job submission

$t0:
\begin{align*}
\text{Dedicated resources pool} & \quad \{ \\
\text{count: 0} & \quad \text{count: 1} \\
\text{Job queue: } & \quad \emptyset
\end{align*}

$t1:
\begin{align*}
\text{new job} & \rightarrow \text{counters increased} \\
\text{Dedicated resources pool} & \quad \{ \\
\text{count: 1} & \quad \text{count: 2} \\
\text{Job queue: } & \quad \text{[Filled]}\end{align*}
Redirection

Overshooting threshold $\rightarrow$ redirect job (in example: $threshold = 3$)

Redirected green job $\rightarrow$ restarted on dedicated resources pool

After redirection $\rightarrow$ Resets counters
Implementation Details

Job counter

- **Redirection threshold** = input parameter
- Multiple jobs overshoot → most resources job redirected

Redirection

- Redirected jobs → submitted to dedicated scheduler
- No further redirection possible

Dedicated resources pool size

- Input parameter: requires tuning
- Job too big → no redirection
Experiments

Simulation:

- **Workload replay: production logs**
  Clusters: Curie, Intrepid, RICC

- 20 weeks extracted per cluster
  Each week has min 70% average load

- **Comparison:** FCFS-Easy VS FCFS-Easy + **Redirection**
  FCFS + Easy backfilling (implemented in real schedulers)

- **Objective:** *bounded slowdown* → user satisfaction

---

1 https://www.cse.huji.ac.il/labs/parallel/workload/logs.html
Metric: Average & Max Bounded Slowdown

Job definition:

Bounded slowdown:

$$BSLD_j = \max \left( \frac{\text{wait}_j + p_j}{\max (p_j, 60)}, 1 \right)$$  \hspace{1cm} (1)
Parameters Tuning

**Dedicated pool size:** $\alpha$  
$(\alpha = 0.1 \rightarrow 10\%)$  

**Redirection threshold:** $\tau$
Online Scheduling with Redirection for Parallel Jobs

Results

**Ratio**: FCFS-Easy + **Redirection** / FCFS-Easy

<table>
<thead>
<tr>
<th>Curie</th>
<th>Intrepid</th>
<th>Ricc</th>
</tr>
</thead>
<tbody>
<tr>
<td>α=0.15</td>
<td>α=0.2</td>
<td>α=0.15</td>
</tr>
<tr>
<td>α=0.2</td>
<td>α=0.25</td>
<td>α=0.2</td>
</tr>
</tbody>
</table>

Lower is better: under black line → redirection improves objective
Conclusion

Redirection algorithm → compatible with other algorithms

No preemption → job killed and restarted

Redirected jobs → Dedicated resources pool

Simulation: redirection improves slowdown
   parameter tuning: redirection threshold & partition size

Future works:

- Dynamic partitioning: variable dedicated pool size
- Load balancing: dedicated pool loaded → disable redirection
- Real scheduler: implementation
- Automating parameters tuning → reinforcement learning
Contacts Informations

Open science:
- **Reproduce**: experiments and visualisations
- **Repository**: https://gitlab.inria.fr/adfaure/evipar

Contact authors at the following addresses:
- adrien.faure@protonmail.com
- giorgio.lucarelli@univ-lorraine.fr
- olivier.richard@imag.fr
- denis.trystram@imag.fr