Revisit the Scheduling Problem with Calibrations

Calibration Minimization and Weighted Throughput Maximization

Lin Chen[1] **Yixiong Gao**[2] Minming Li[2] Guohui Lin[3] Kai Wang[2]

[1] Zhejiang University [2] City University of Hong Kong [3] University of Alberta

17th Workshop on Models and Algorithms for Planning and Scheduling Problems, MAPSP 2026

Introduction

Model

- There are m identical parallel machines.
- Each job j has **unit** processing time and one feasible window:

$$r_j < t_j \leq d_j.$$

- Time slot t denotes the interval $(t-1, t]$.

Calibration

A calibration started at time s makes one machine available for the next T time slots.

A job may be processed only while its assigned machine is calibrated.

Calibration Minimization

- Schedule all jobs
- Minimize the number of calibrations

Weighted Throughput Maximization

- Use at most k calibrations
- Maximize the total weight of completed jobs

Related Work

Year	Authors	Contribution
2013	M. A. Bender et al.	Introduced the standard offline minimization model; **Lazy-Binning** is optimal on one machine, gives a 2-**approximation** on identical parallel machines, and is exact when all deadlines are distinct.
2019	V. Chau et al.	Introduced weighted throughput maximization under a budget of K calibrations; obtained a $1/3$-**approximation** for unit-time jobs.
2022	Z. Chen and J. Zhang	For the online $m = \infty$ model with calibration time λ, gave a $(3(e+1)\lambda + 3e + 7)$-**competitive** algorithm and a lower bound of $\max\{e, \lambda\}$; when $\lambda = 0$, the ratio is $3e + 7 \approx 15.16$.
2023	E. Sun et al.	Improved the same online $m = \infty$ setting (equivalently, online rent minimization) to 6-**competitive** for $\lambda = 0$ and $6(\lambda + 1)$-**competitive** when each calibration has delay λ.
2025	V. Chau et al.	For the broader multi-interval model, obtained a tight $(1 - 1/e)$-**approximation** for weighted maximization.

Our Results

Calibration Minimization

Assumption	Guarantee	Running time
m is constant	exact algorithm	$O(n^{8+6m})$
T is constant	exact algorithm	$O(n^8 m^{3T})$
m, T are inputs	$(1+\varepsilon)$-approximation (PTAS)	$O(n^{17+18\lceil 1/\varepsilon \rceil})$

Weighted Throughput Maximization

Assumption	Guarantee	Running time
m is constant	exact algorithm	$O(k^2 n^{8+6m})$
T is constant	exact algorithm	$O(k^2 n^8 m^{3T})$
m, T are inputs	$(1-\varepsilon)$-approximation (PTAS)	$O(n^{19+18\lceil 1/\varepsilon \rceil})$

The weighted version uses the same dynamic programs, with rejection and a calibration-budget coordinate.

Common Structure

Canonical Schedules

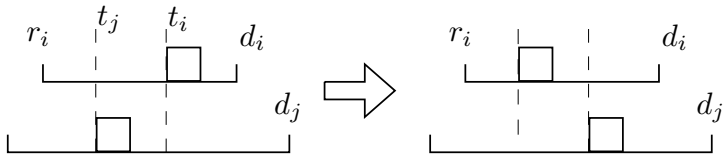
- Sort jobs by nondecreasing deadlines, breaking ties by release times.
- Index jobs from 1 to n after sorting.
- For either objective, after fixing the completed set, an EDF exchange gives

$$i < j,\ r_i \le t_j \quad \Longrightarrow \quad t_i \le t_j.$$

- Calibration starts and job completion times can be restricted to polynomial candidate sets:

$$\Psi = \{d_j - s : j \in J,\ s = 0, \dots, n\},$$

$$\Phi(j) = \{t \in \Psi : r_j < t \le d_j\}.$$



The exchange swaps job identities; the selected set and total weight are preserved.

The Interval Split

Subproblem jobs.

For $t_1, t_2 \in \Phi$,

$$J(j, t_1, t_2) = \{i \in J : i \leq j,\ r_i \in [t_1, t_2)\}.$$

- Place job j at a candidate slot t.
- If jobs in $J(j, t_1, t_2)$ are only scheduled during $[t_1, t_2)$:
  - By EDF, jobs in $J(j-1, t_1, t)$ must be scheduled during $[t_1, t)$.
  - By release times, jobs in $J(j-1, t, t_2)$ must be scheduled during $[t, t_2)$.
- Thus we can partition all smaller-index jobs and the time interval simultaneously into:

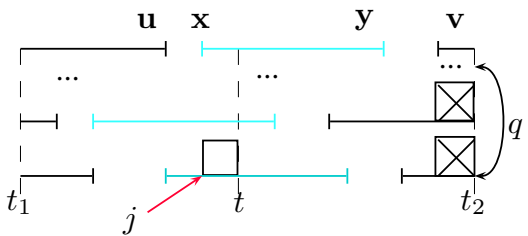
$$J(j-1, t_1, t) \quad \text{and} \quad J(j-1, t, t_2).$$

Dynamic Programming Design

Generic state.

$$(j, t_1, t_2, q, B^-, B^+)$$

Here B^-, B^+ encode boundary calibrations, and q reserves capacity at the right boundary.



When a slot first becomes active, enumerate its covering calibrations.

Two Exact Representations

Fixed m: machine-indexed

At each boundary, store the start time of the calibration on each machine:

$$\mathbf{u}, \mathbf{v} \in \left(\{\text{null}\} \cup \Psi\right)^m.$$

This is polynomial when m is fixed.

Fixed T: lifetime counts

At time t, store only how many calibrations have each remaining lifetime:

$$\mathbf{c}_t = (c_{t,1}, \ldots, c_{t,T}) \in [m]^T.$$

This is polynomial when T is fixed.

The recursive split is identical; only the boundary descriptor changes.

From Minimization to Weighted Maximization

Budgeted weighted state

$$F(j, t_1, t_2, q, B^-, B^+, b)$$

is the maximum schedulable weight inside the subproblem using exactly b additional calibrations.

New decisions

- reject job j;
- if job j is scheduled, add w_j;
- maximize total weight instead of minimizing calibrations.

Budget split

For a split descriptor X, $b_L + b_R + \kappa(X) = b$.

Here b_L, b_R are the left/right recursive budgets;

$\kappa(X)$ counts calibrations introduced by X.

Enumerating (b_L, b_R) adds a factor k^2 to the running time.

Compression

Why Compression Is Needed

- When both m and T are inputs, neither exact representation is polynomial.
- The obstacle is local diversity: too many distinct calibration starts can cover the same slot.
- The PTAS creates a compressed schedule in which every active slot sees only

$$h = 2\lceil 1/\varepsilon \rceil + 1$$

distinct calibration starting and ending times.

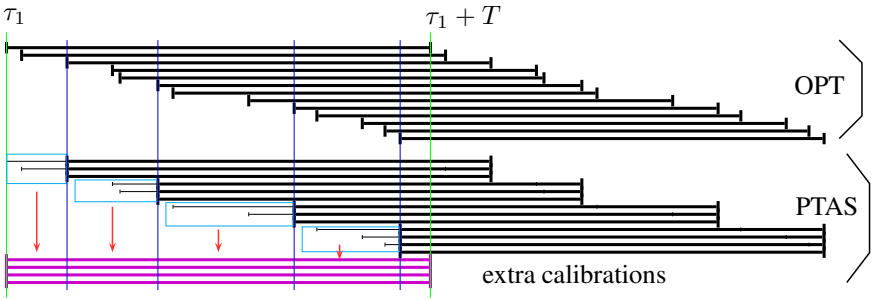
Idea

Transform an **optimal** solution into a **compressed** solution.

Keep every job in the same time slot; change only how calibrations support jobs.

Block Transformation

- Group consecutive calibrations whose starts lie in a window of length T.
- Split a block of ℓ calibrations into $\lceil 1/\varepsilon \rceil$ consecutive groups.
- Delay all calibrations in each group to its latest start time.
- Add $\lfloor \varepsilon \ell \rfloor$ extra calibrations to cover affected jobs.



Configuration DP

Configuration

A configuration records the compressed local picture:

$$C = \langle \alpha, \eta \rangle,$$

where α_i is a calibration start time and η_i is its multiplicity.

- At most h distinct starts are stored.
- Boundary configurations are merged when a time slot becomes active.
- Invalid merges are discarded if they exceed h starts or more than $m + \lfloor \varepsilon m \rfloor$ simultaneous calibrations.
- For fixed ε, the number of configurations is polynomial.

Back to the Original Resources

Remove the extra machines

Delay overloaded calibrations until at most m machines are active in every slot.

This preserves the job schedule and the number of calibrations.

Restore the budget (for the maximization version only)

The weighted PTAS first allows $\bar{k} = k + \lfloor \varepsilon k \rfloor$ calibrations.

A final trimming step restores the strict budget k (from a PTAS solution using C calibrations):

- Charge every completed job to the calibration that covers it.
- Let $L(c)$ be the total charged weight of calibration c.
- If $C > k$, delete the $C - k$ calibrations with smallest charged load.
- Delete only the jobs charged to those calibrations.

Future Work

- The computational complexity of this problem is still open.
- The PTAS relies on the unit processing times. With non-unit processing times, the EDF split and the compression argument do not directly extend. Can we find new structural tools for these broader calibration models?

Thank you!