CATIOS: Time-Resolved I/O-Aware Job Scheduling for HPC Systems

YuTsen Tseng[1], Masatoshi Kawai[1], Keichi Takahashi[2], and Hiroyuki Takizawa[1]

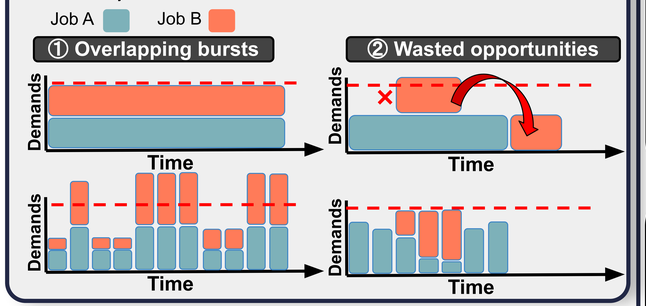
Tohoku University[1], University of Osaka[2]

Background

- HPC job scheduling coordinates resources to maximize system utilization.
- Traditional schedulers focus on computing resources, yet workloads are increasingly becoming data-intensive.
  - ➢ Bandwidth constrains the job performance.
- I/O-aware job scheduling mitigates contention by considering the I/O demand of a job.

Problem Statement

- Existing I/O-aware schedulers [1, 2] use average I/O bandwidth or simple periodic I/O models.
- Average I/O bandwidth hides I/O bursts that actually drive contention.



Proposed Solution

- Contention-Avoiding Temporal I/O-aware job Scheduling (CATIOS)
  - ➢ Model the temporal I/O behavior of jobs, simulate system-wide I/O demand over time, and make scheduling decisions accordingly.
  - ➢ Avoids peak-overlap contention and fills idle intervals, addressing average-based blind spots.
- Configurable scheduling policy
  - ➢ Adapt to admin goals on heterogeneous workloads, balancing fairness and throughput.
- Job-script-only input
  - ➢ Reduce user burden through leveraging semantic analysis to retrieve historically similar jobs and infer job behavior.

The Design of CATIOS

Command-based Job Matching

- Retrieve similar historical jobs
  - ➢ Embed the job's command line with Sentence-BERT, selected for its efficiency and robustness, and compare it with that of historical jobs.
  - ➢ Find multiple semantically similar historical jobs and select the one with the closest processor count to ensure the execution scale is comparable.

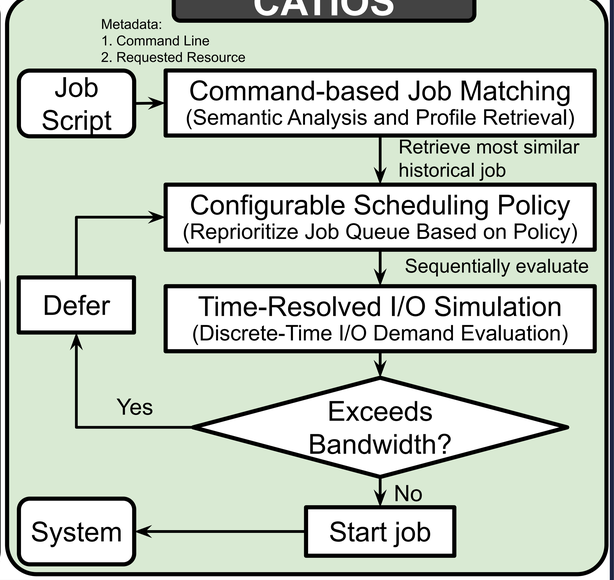
Configurable Scheduling Policy

- Scheduling policies
  - ➢ FCFS (fairness-oriented), LJF (throughput-oriented)
  - ➢ Mix (tune α, β to trade fairness and throughput.)

$$P(j) = \alpha \cdot \text{norm_wait}(j) + \beta \cdot \text{norm_runtime}(j)$$

- Queue action
  - ➢ Sort pending jobs by priority and pass them to I/O simulation in that order.
  - ➢ On rejection, defer the job and evaluate the next; periodically recompute priorities.

CATIOS

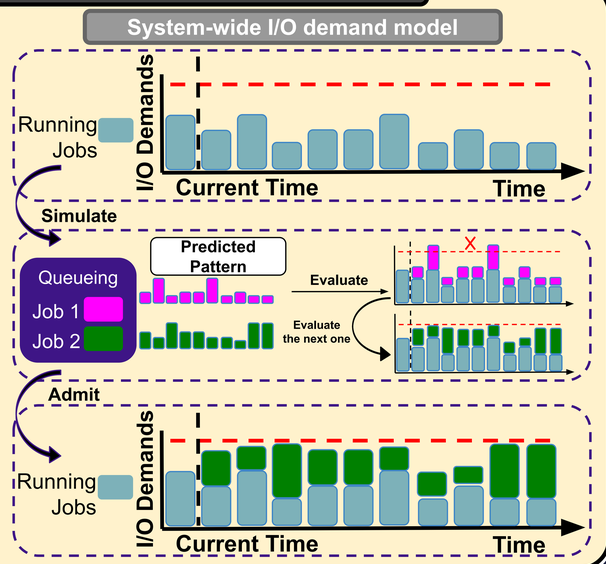


Time-resolved I/O Simulation and Job Execution Decision

- System-wide I/O demand model
  - ➢ Maintain total I/O demand within storage bandwidth at every moment.
  - ➢ Given the start time t_0 of the candidate job, its per-second I/O demand D is summed with the total I/O demand of running jobs *load* to estimate the aggregate demand A.

$$A[t] = \text{load}[t] + D[t - t_0].$$

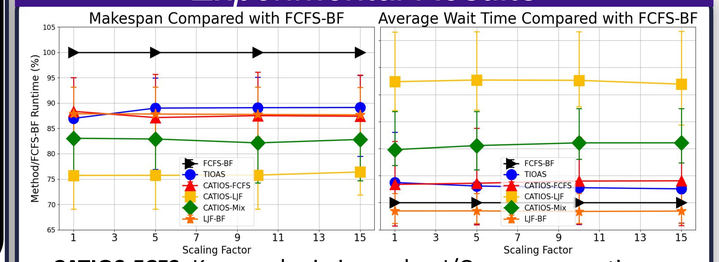
- Resource availability check
  - ➢ Check compute and I/O resource availability for $[t_0, t_0+n-1]$.
  - ➢ For all $t \in [t_0, t_0+n-1]$: $A[t] \leq B$ (bandwidth).
  - ➢ Robustness: clip any $D[i]$ to $0.75 \cdot B$ to limit outlier spikes.
- Job execution decision
  - ➢ If resources are available, start job, update total I/O demand, and resource usage.
  - ➢ If resources are unavailable, defer job execution and check the next job.



Experimental Setup

- Dataset: Reproduce Blue Waters traces [3] via Darshan log [4], converted to per-second I/O profiles.
- Simulator: SimGrid [5] extended to model time-resolved I/O with bandwidth and buffer constraints.
- Performance metrics: Makespan and average wait time, normalized to FCFS-BF.
- Compared methods
  - ➢ FCFS and LJF with backfilling (BF)
  - ➢ Traditional I/O-Aware job Scheduling (TIOAS)
  - ➢ CATIOS variants: CATIOS-FCFS/LJF/Mix.

Experimental Results



- **CATIOS-FCFS**: Keeps submission order; I/O-aware execution decision avoids overlapping I/O bursts; more consistent gains than TIOAS as load rises.
- **CATIOS-LJF**: Best throughput and strongest makespan improvement; higher wait for short jobs.
- **CATIOS-Mix** (α:β=0.95:0.05): Tunable balance; improves makespan while moderating wait growth under contention.

Conclusion and Future Work

- CATIOS mitigates I/O contention and reduces makespan while keeping average wait times controlled compared to static approches, achieving a trade-off ratio of approximately 1:1.3 between makespan reduction and average wait time increase.
  - ➢ CATIOS effectively alleviates I/O contention even in I/O-intensive scenarios, showing its capability to improve data-intensive HPC systems.
  - ➢ CATIOS' configurable scheduling policies allow admin to balance fairness and throughput as objectives evolve.
- Future work: extend to multi-resource co-scheduling and improve profile matching accuracy.

This work was supported by the Council for Science, Technology and Innovation (CSTI), Cross ministerial Strategic Innovation Promotion Program (SIP), "Development of a Resilient Smart Network System against Natural Disasters" Grant Number JPJ012289 (Funding agency: NIED), by JSPS KAKENHI Grant Number JP24K2945, and by Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures (JHPCN) in Japan (Project ID: jh250062).

[1] AI4IO: Wyatt M. R.; Herbein S.; Gamblin T.; Taufer M. Int. J. High Perform. Comput. Appl. 36(3) (2022) 370–387. doi:10.1177/10943420221079765
[2] Periodic I/O scheduling for supercomputers: Guillaume A.; Ana G.; Valentin L. F. Proc. HPDC '21, ACM, pp. 141–152, 2021. doi:10.1145/3431379.3460653
[3] Blue Waters Darshan data: National Center for Supercomputing Applications. Blue Waters User Portal. https://bluewaters.ncsa.illinois.edu/data-sets
[4] Darshan tool: Snyder S.; Carns P.; Harms K.; Ross R.; Lockwood G. K.; Wright N. J. ESPT 2016, 9–17. doi:10.1109/ESPT.2016.006
[5] SimGrid: Casanova H.; Giersch A.; Legrand A.; Quinson M.; Suter F. Parallel Computing 123 (2025) 103125. doi:10.1016/j.parco.2025.103125

(URL) https://www.hpc.is.tohoku.ac.jp
(E-mail) tseng.yu.tsen.s4@dc.tohoku.ac.jp