University of Basel

DMI HPC

Improving Job Throughput in HPC with Adaptive Time Limit Management

Thomas Jakobsche **University of Basel, Basel, CH**

Florina M. Ciorba University of Basel, Basel, CH
Jim Brandt Sandia National Laboratories, US
Ann Gentile Sandia National Laboratories, US
Quentin Guilloteau University of Basel, Basel, CH
Michael Ott Leibniz Supercomputing Centre, DE
Osman Seckin Simsek University of Basel, Basel, CH
Torsten Wilde Hewlett Packard Enterprise, DE

SLUG'24 - Slurm User Group Meeting 2024
University of Oslo, Problemveien 11, 0313, Oslo
11-13 September 2024

Sandia National Laboratories

Hewlett Packard Enterprise

lrz Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities

Outline

Context

Dagstuhl Seminar 23171

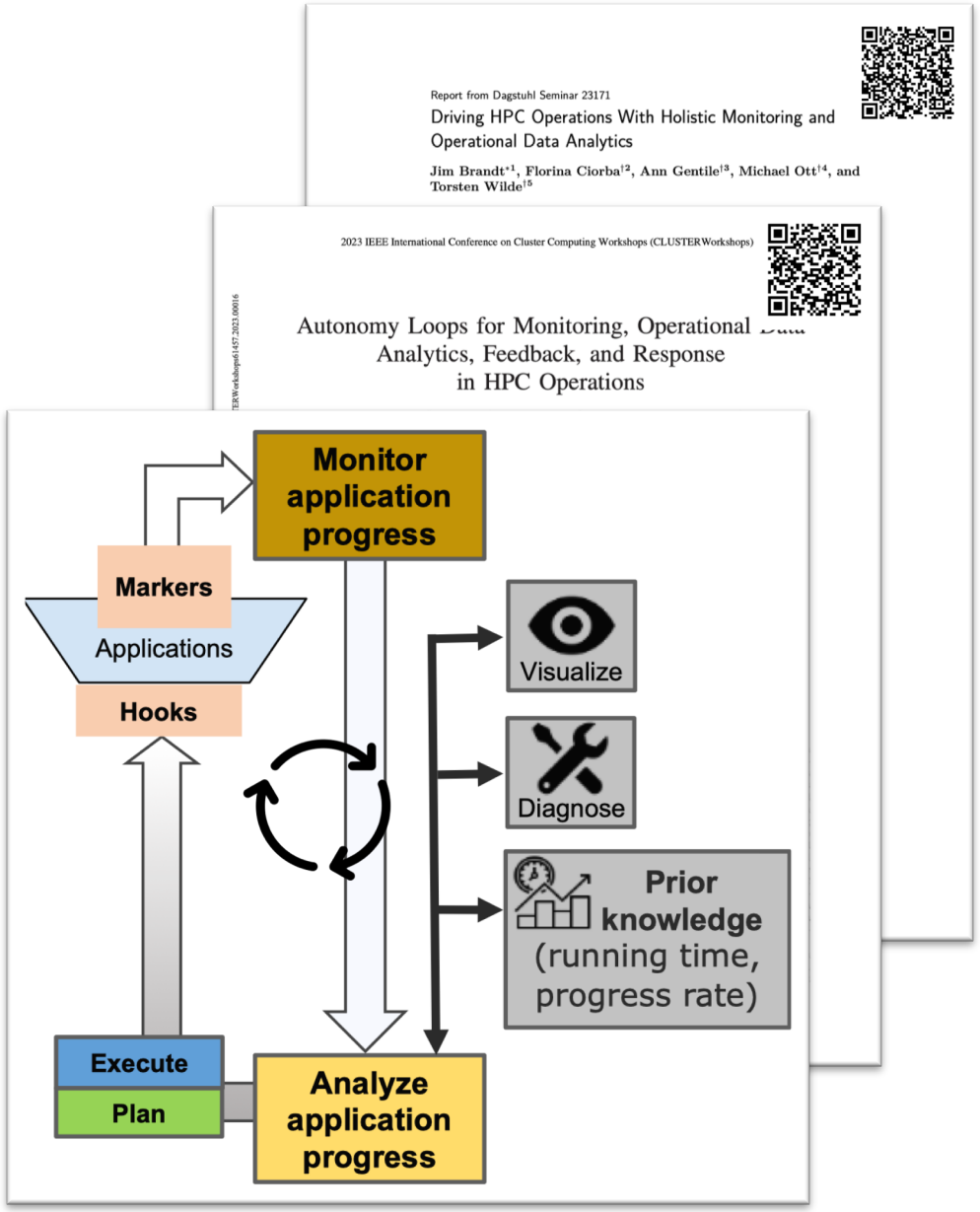
- HPC monitoring is well explored and generates tons of data
- Analysis of and response to monitoring data is mostly manual
- Human-in-the-loop is becoming intractable and unfeasible

➢ Wonderful And Fundable Vision Report (WAFVR)

WAFVR Initiative

- Use Cases of Autonomy Loops for HPC Operations
- Automated response with **human-on-the-loop** instead of simply human-in-the-loop

➢ **Adapting time limits of running jobs to improve system throughput and reduce wasted time and energy**



Problem and Solution

Problem

Wasted energy and computational time

- Losing computational progress to **timeouts**
- Inefficient scheduling due to time limit **overestimation**

Motivation

Ensure the efficient use of HPC resources and reduce energy consumption

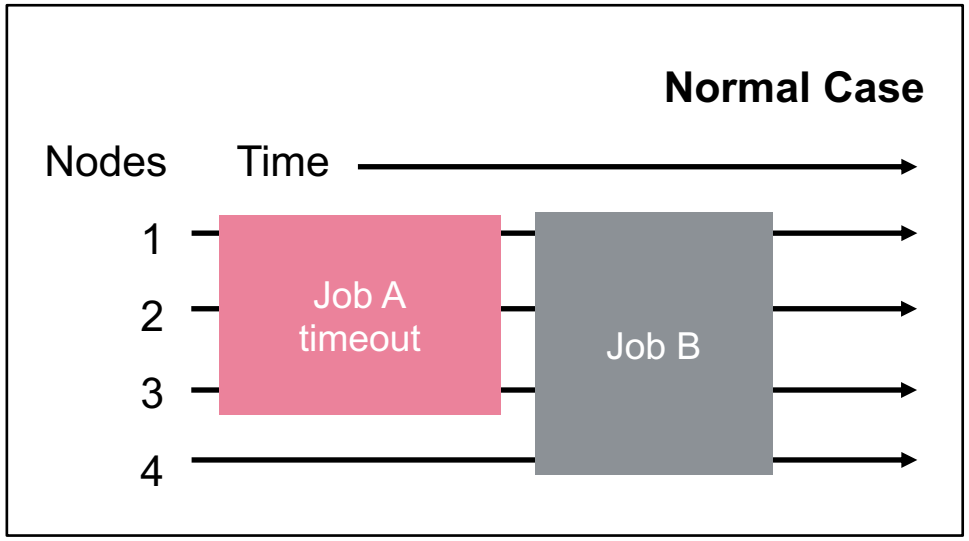
- **Scientific:**
  Improve science / second
- **Environmental:**
  Reduce wasted energy and computations

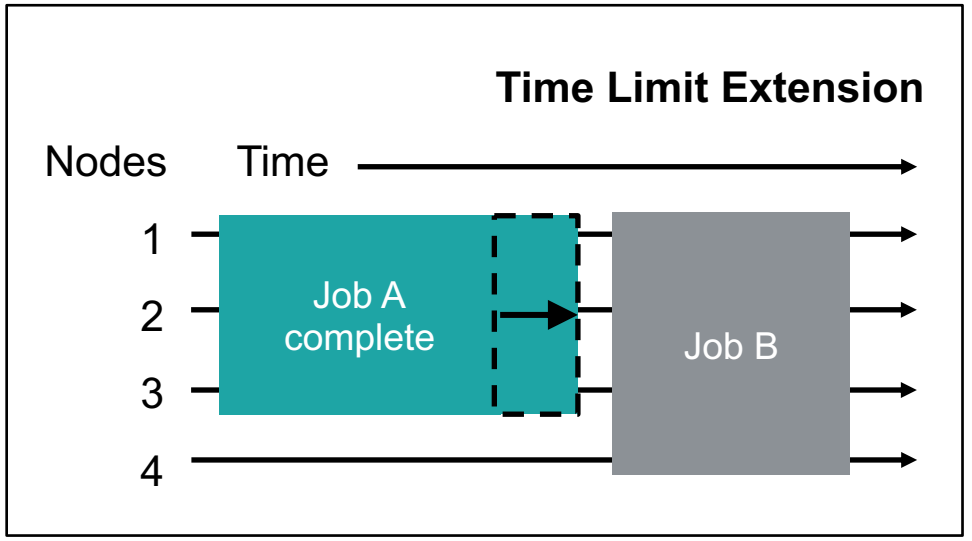
Proposed Solution

Avoid wasteful timeouts through autonomy loop between system scheduler and applications

- “Blanket” OverTimeLimit vs. **individual and informed adjustments**
- Adjust job time limits **based on application progress** towards checkpoints or completion

Challenges of OverTimeLimit and Time Limit Adjustments

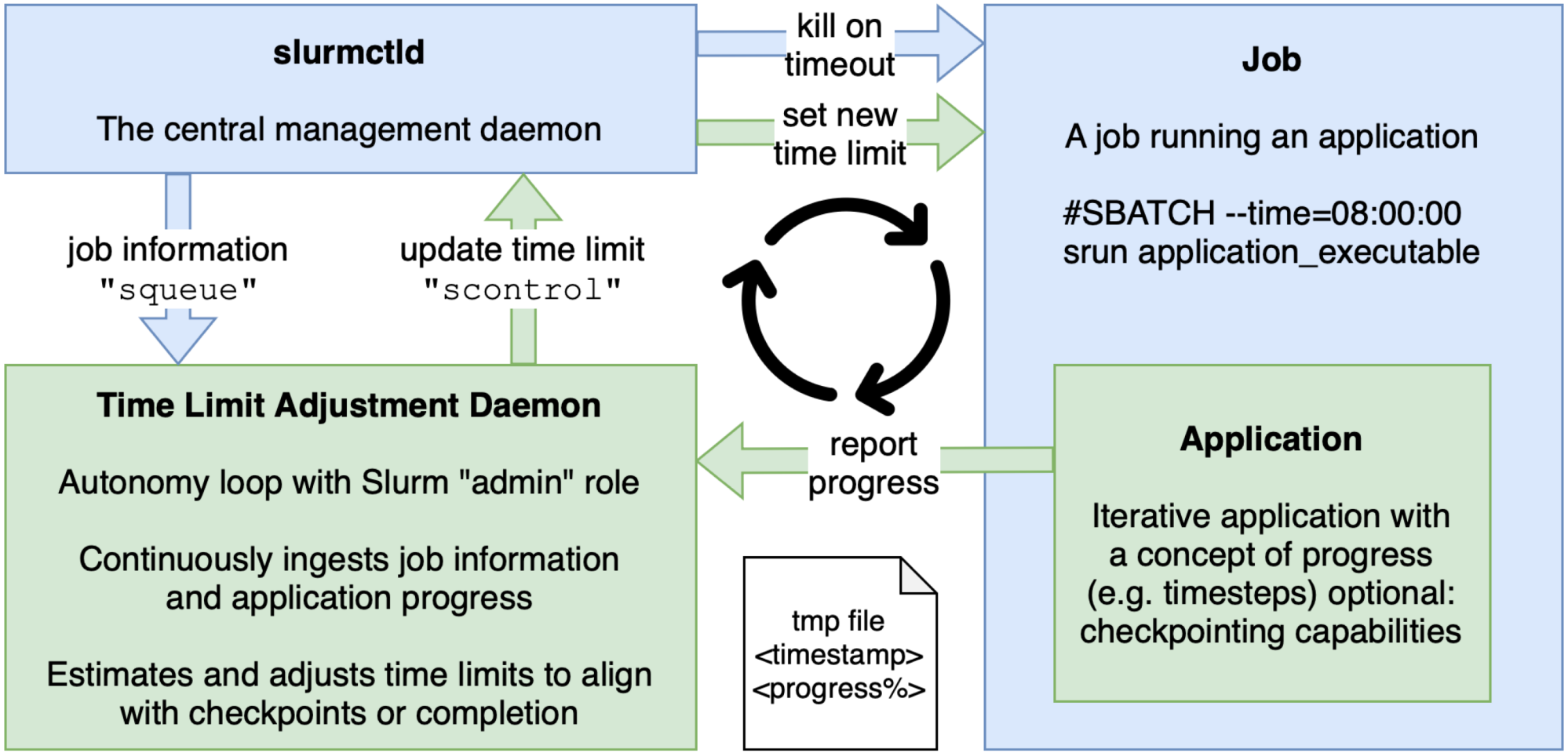




- Time limit extensions of executing jobs can avoid timeouts but delay queued jobs
- Time limit extensions of backfilled jobs can delay higher priority jobs (soft vs. hard time limits)
- “Blanket” time limit extension of all jobs through OverTimeLimit does not guarantee completion

➢ **Our approach incorporates application progress for an individualized job time limit adjustment**

Proof-of-Concept
Time Limit Adjustment



Proof-of-Concept
Time Limit Adjustment: Experimental Setup

System Specifications

- 20-node fully-controlled research cluster
- Isolated experiments through reservations

Target Application

- SPH-EXA simulation framework executing a Sedov blast simulation
- Extended with progress reporting

Background Jobs

- Sleep jobs with configurations from a random subset of short production jobs* from Aug'23
- Requested nodes (1-6), execution times (1m-10m), time limits (10m-1w), + submit pattern

Workload and Experiments

Workload of 46 jobs

- 39 background jobs (all completing)
- 7 randomly injected identical SPH-EXA Sedov jobs, submitted with 7m time limit but need ~8m to complete

Experiments with the entire 46-job workload

1) SPH-EXA timeout
2) SPH-EXA timeout + resubmission with 30m
3) SPH-EXA time limit extension, no timeout

Research Questions

- Do adjustments "break" job scheduling?
- Do adjustments reduce wasted time & energy?

*sciCORE, scientific computing center at University of Basel, Switzerland

Proof-of-Concept
Time Limit Adjustment: Results: CD - Completed, TO - Timeout

Workload & Scheduling Characteristics	1) Timeout average of 5 repetitions max variation ±1.8%			2) Timeout + Resubmission average of 5 repetitions max variation ±6.3%			3) Time limit adjustment average of 5 repetitions max variation ±1.9%		
	CD	**TO**	**Total**	**CD**	**TO**	**Total**	**CD**	**TO**	**Total**
Number of jobs	39	7	46	**46**	**7**	53	**46**	**0**	46
CPUTime [h]	333	95	428	443	**95**	538	441	**0**	441
ConsumedEnergy [MJ]	6.6	2.2	8.8	9.0	**2.2**	11,2	9.0	**0**	9.0
Throughput [Compl. jobs/h]	55			**53**			**62**		
Average job wait time [s]	317			**419**			**332**		

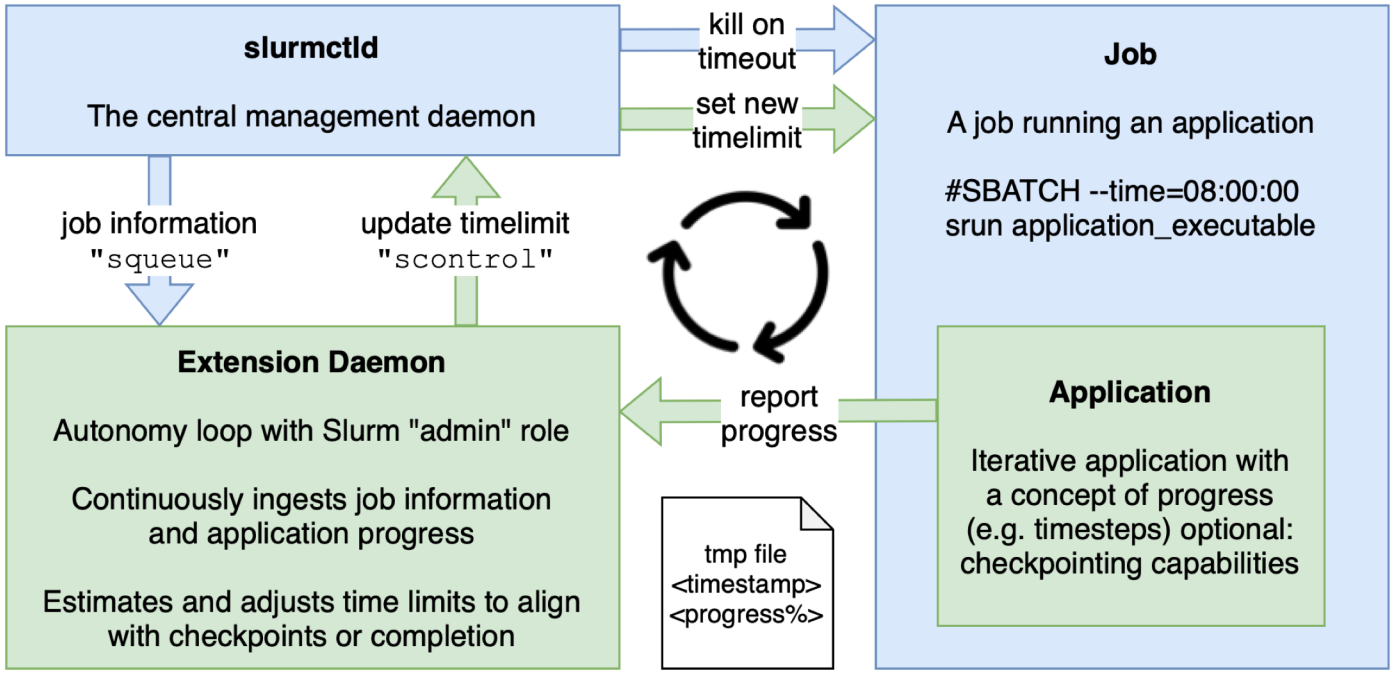
Conclusion and Next Steps

Conclusion

- Working autonomy loop to dynamically adjust time limits
- **Tradeoff:** Slightly increased average job wait time for achieving higher throughput

Next Steps

- Testing on production HPC systems
  - Can we test on your system ☺?
- Aligning time limit adjustments with application checkpoints



Questions for the Slurm Community

Application Developers

- Are they willing to implement progress reporting? [<timestamp>, <progress%>]

System Administrators

- Have they tested similar approaches? Do they work?
- Is there a need for automated decisions?

Slurm Developers

- Are there plans to develop communication between the scheduler and applications?
- Are there plans to incorporate monitoring/application data for scheduling decisions?

Come talk to me in person or contact me & co-authors at: thomas.jakobsche@unibas.ch (and title slide)

University of Basel

DMI HPC

Improving Job Throughput in HPC with Adaptive Time Limit Management

Thomas Jakobsche **University of Basel, Basel, CH**

Florina M. Ciorba University of Basel, Basel, CH
Jim Brandt Sandia National Laboratories, US
Ann Gentile Sandia National Laboratories, US
Quentin Guilloteau University of Basel, Basel, CH
Michael Ott Leibniz Supercomputing Centre, DE
Osman Seckin Simsek University of Basel, Basel, CH
Torsten Wilde Hewlett Packard Enterprise, DE

SLUG'24 - Slurm User Group Meeting 2024
University of Oslo, Problemveien 11, 0313, Oslo
11-13 September 2024

Sandia National Laboratories

Hewlett Packard Enterprise

lrz Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities