Exploring Distributed HPC Scheduling with Randomized Resource Stealing

Kiran Ramamurthy
Dept. of Computer Science
Illinois Institute of Technology

Dr. Ioan Raicu
Dept. of Computer Science
Illinois Institute of Technology

Abstract

- Efficiently scheduling large number of jobs over large scale distributed systems is very critical.
- Today’s state-of-the-art job schedulers mostly follow a centralized architecture that is master/slave architecture.
- Aims at providing HPC support on top of MATRIX MTC framework.
- In-corporates resource stealing with work stealing.

Working

- Scheduler chooses multiple nodes in random.
- Requests for resource information on the nodes.
- Validates if sufficient resources are available to complete the tasks.
- If Yes, Breaks the task into sub-tasks and migrates it to the nodes selected.
- Source receive the results after execution.

Architecture

Execution Unit

MATRIX-HPC Preliminary Results

Conclusion

- HPC support for MATRIX currently out performs SLURM++ for small task size
- With medium task size, HPC support for MATRIX performs better at 50 nodes.
- With large task size, where each task needs 20 needs HPC support for MATRIX performs better at 50 nodes in comparison to SLURM++ at 100 nodes,
- The SLURM++ needs to be run at lesser scales to make a better comparison.

Future Work

- Comparison to be made with SLURM++ at higher scales of 100 nodes and above
- Random neighbor selection will be replaced with history based neighbor selection.
- Resource information can be efficiently maintained by incorporating a key-value pair. This keeps the resource information of each node up to date with less operations.

References
