TetriSched: Global Rescheduling with Adaptive Plan-ahead in Dynamic Heterogeneous Clusters

Alexey Tumanov*, Timothy Zhu*, Jun Woo Park*, Michael A. Kozuch**, Mor Harchol-Balter*, Greg Ganger*
*Carnegie Mellon University, **Intel Labs

Background

- Clusters are increasingly heterogeneous
  - Resource types: GPUs, FPGAs, large RAM
  - Topology: rack locality, failure domains, loaded data
- Workloads vary in time and resource needs
  - E.g., best-effort analytics vs. SLO jobs w/ deadlines
  - E.g., 2hrs on 5xGPU or 4hrs on 10xCPU
- Current schedulers don’t exploit this flexibility well
  - Results: wasted resources, missed deadlines, high latency

Problem Statement

- Heterogeneity results in many placement options
  - Which resources/types to allocate? (space)
  - Run now or wait for better resource? (time)
- Key challenges
  - Express and quantify combinatorially many options
  - Leverage runtime estimates robustly
  - Exploit this knowledge to improve allocation efficiently

TetriSched System Architecture

- Space-time request language
  - [R1] space-time constraint awareness
  - [R2] soft constraints (preference) awareness
  - [R3] combinatorial constraints
  - [R4] gang scheduling
  - [R5] composability for global scheduling

Experimental Results

- Real Cluster: 256 nodes
  - Workload: FB2009 SLO + Yahoo BE (SWIM)
  - Rayon/TetriSched >> Rayon/CapacitySched
- Real Cluster: 80 nodes
  - Workload: synthetic GPU + MPI + BE
  - Soft constraints: 2x perf boost
  - Plan-ahead + global scheduling: 2.5x performance boost over baseline

Real Cluster: 80 nodes
- Workload: synthetic GPU + MPI + BE
- Soft constraints: 2x perf boost
- Plan-ahead + global scheduling: 2.5x performance boost over baseline

Real Cluster: 80 nodes
- Workload: synthetic GPU + MPI + BE
- Plan-ahead + global scheduling: 2.5x performance boost over baseline