FLEXRR: SOLVING THE STRAGGLER PROBLEM FOR ITERATIVE CONVERGENT PARALLEL ML
Aaron Harlap, Henggang Cui, Wei Dai, Jinliang Wei, Greg Ganger, Phillip B. Gibbons*, Garth Gibson, Eric Xing
Carnegie Mellon University, *Intel Labs

PARALLEL ML
• Input (training) data spread among workers
  › Workers compute adjustments to model params
• Synchronize progress occasionally
  › BSP: barrier synch each clock (iteration)
  › SSP: bounded number of clocks apart

• Stragglers are common in practice
  › One worker slower than others
  › Long-term: load imbalance
  › Transient: short-term slow-down
    • E.g., garbage collection, stop condition check, resource contention, etc.

SLACK + RAPID REASSIGNMENT
• Goal: never reach the slack boundary
• Approach
  › Detect slowed workers quickly
  › Shift some work to faster workers

<table>
<thead>
<tr>
<th>Initial Work Assignments</th>
<th>Rebalanced Work Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worker 1</td>
<td>Worker 2</td>
</tr>
<tr>
<td>Worker 1</td>
<td>Worker 2</td>
</tr>
</tbody>
</table>

• Challenges
  › Detecting and reacting quick enough
  › Limiting overhead
  › Local State

FLEXRR IMPLEMENTATION
• Integrated into LazyTable system
• Used with CF, TM, and MLR applications
• Need both SSP and Rapid-Reassignment
  › Each solution on its own only partially solves the straggler problem

EFFECT OF STRAGGLERS
• Fast workers wait for slow workers
• BSP: wait at each barrier synch
  › ... for slowest worker in each clock
• SSP: can mitigate short transient effects
  › ... but not ones beyond the slack allowed

RAPID-REASSIGNMENT PROTOCOL
• Each worker has designated group of helpers.
  › Bounds overhead as scale increases
• Workers multicast when "nearly done"
• Workers compare messages to own progress
• If behind, re-assign some work
  › Local state recomputed if necessary (TM)
• Once help begins, workers re-assign more work

RESULTS
• Ran on AWS, using 64 8 core machines
• Even with no delays introduced(X=0), 35% speedup
• Ran on Dedicated Cluster of 16 8 core machines
  › More controlled environment
• Big improvements when delays are introduced
  › Emulating straggler scenarios of varied intensity
• Even for TM, where local state adds overhead,
  FLEXRR outperforms SSP and BSP RR