Aggressive Cloning of Jobs for Effective Straggler Mitigation

Ganesh Ananthanarayanan, Ali Ghodsi, Scott Shenker, Ion Stoica
Small jobs increasingly important

• Most jobs are **small**
  – 82% of jobs contain less than 10 tasks (Facebook’s Hadoop cluster)

• Most small jobs are **interactive** and **latency-constrained**
  – Data analyst testing query on small sample

• Small jobs particularly sensitive to **stragglers**
Straggler Mitigation

• **Blacklisting:**
  – Clusters periodically diagnose and eliminate machines with faulty hardware

• **Speculation**: **Non-deterministic** stragglers
  – Complete systemic modeling is intrinsically complex [e.g., Dean’12 at Google]
    – **LATE** [OSDI’08], **Mantri** [OSDI’10]...
Despite the mitigation techniques...

- **LATE**: The slowest task runs 8 times slower than the median task.

- **Mantri**: The slowest task runs 6 times slower than the median task.

  - (...but they work well for large jobs)
State-of-the-Art Straggler Mitigation

**Speculative Execution:**
*(in LATE, Mantri, MapReduce)*

1. **Wait**: observe relative progress rates of tasks

2. **Speculate**: launch copies of tasks that are predicted to be stragglers
Why doesn’t this work for small jobs?

1. Consist of just a **few tasks**
   - Statistically hard to predict stragglers
   - Need to **wait longer** to accurately predict stragglers

2. Run all their tasks **simultaneously**
   - Waiting can constitute **considerable fraction** of a small job’s duration

*Wait & Speculate* is ill-suited to address stragglers in small jobs
Cloning Jobs

- **Proactively** launch clones of a job, just as they are submitted
- Pick the result from the *earliest* clone
- *Probabilistically mitigates stragglers*

- Eschews waiting, speculation, causal analysis...

*Is this really feasible??*
Low Cluster Utilization

• Clusters have median utilization of under 20%
  – Provisioned for (short burst of) peak utilization

• Cluster energy-efficiency proposals
  – Not adopted in today’s clusters! 😞
  – Peak utilization decides half the energy bill
  – Hardware and software reliability issues...
Tragedy of commons?

- If every job utilizes the “lowly utilized” cluster...
  - Instability and negative performance effects

*Power-law:*  
- 90% of jobs use 6% of resources  
- FB, Bing, Yahoo!

Can clone small jobs with few extra resources
Strawman

- Easy to implement
- Directly extends to any framework

Job

Earliest
Number of map clones

- Contention for input data by map task clones
- Storage crunch → Cannot increase replication

>> 3 clones
Task-level Cloning

- Job
  - M1
  - Earliest
  - M1
  - Earliest
  - M2
  - Earliest
  - M2
  - Earliest
  - R1
  - Earliest
  - R1
≤3 clones suffices

Strawman

Task-level Cloning
Dolly: Cloning Jobs

- **Task-level** cloning of jobs
- Works within a **budget**
  - Cap on the extra cluster resources for cloning
Evaluation

• Workload derived from Facebook traces
  – FB: 3500 node Hadoop cluster, 375K jobs, 1 month

• Trace-driven simulator

• Baselines: LATE and Mantri, + blacklisting

• Cloning budget of 5%
Baseline: LATE

Small jobs benefit significantly!

Average completion time improves by 44%
Baseline: Mantri

Small jobs benefit significantly!

Average completion time improves by 42%
Intermediate Data Contention

• We would like every reduce clone to get its own copy of intermediate data (map output)

• Not replicated, to avoid overheads

• "What if a map clone straggles?"
Intermediate Data Contention

Wait for exclusive copy or contend for the available copy?
Conclusion

• **Stragglers in small jobs** are not well-handled by traditional mitigation strategies
  – Guessing task to speculate very hard, waiting wastes significant computation time

• **Dolly: Proactive** Cloning of jobs
  – **Power-law** → Small **cloning budget (5%)** suffices
  – Jobs improve by at least **42%** w.r.t. **state-of-the-art** straggler mitigation strategies

• **Low utilization + Power-law + Cloning?**