On the diversity of cluster workloads and its impact on research results

George Amvrosiadis, Jun Woo Park, Greg Ganger, Garth Gibson,
Elisabeth Baseman, Nathan DeBardeleben

www.pdl.cmu.edu/ATLAS

Carnegie Mellon Parallel Data Laboratory

Los Alamos National Laboratory

Sources for cluster traces today

- Parallel Workload Archive (1993 – 2015)
  - 38 HPC cluster traces
    (each: 1K+ cores, months long)
  - Publications: 250+
- Google cluster trace (2011)
  - 29 days of a 12,000-node cluster
  - Publications: 450+

26	LLNL Atlas	Nov 2006	Jun 2007
27	LLNL Thunder	Jan 2007	Jun 2007
28	ANL Intrepid	Jan 2009	Sep 2009
29	MetaCentrum	Dec 2008	Jun 2009
30	PIK IPLEX	Apr 2009	Jul 2012
31	RICC	May 2010	Sep 2010
32	CEA CURIE	Feb 2011	Oct 2012
33	Intel NetBatch pool A	Nov 2012	Dec 2012

Google cluster-usage traces: schema

Charles Reiss, John Wilkes, Joseph Hellerstein
Version of 2013-05-06, for trace version 2. Revised 2014-11-17
Status: exported outside Google.
Copyright © 2011 Google Inc. All rights reserved.

Google trace: exceedingly popular, but how representative of other clusters?

Project Atlas

- Mandate: use historical data to improve cluster efficiency
  - LANL: scheduler logs, sensor data, OS logs, … → TBs / day
  - Recently: data from Two Sigma, Pittsburgh Supercomputing Center

TWO SIGMA PITTSBURGH SUPERCOMPUTING CENTER

Current goals:

- Investigate overfitting to existing traces in systems literature
- Produce generalizable models of cluster workloads
- Create trace repository and make data publicly available

Atlas repository: current traces

- Two Sigma business analytics clusters: 9 months (2016-2017)
  - 1300 nodes, 31500 cores, 328TB RAM

Entire cluster lifetime

- LANL Mustang general-purpose cluster: 5 years (2011-2016)
  - 1600 nodes, 38400 cores, 100TB RAM
- LANL *Open*Trinity capability cluster: 3 months (2017)
  - Trinity phase 1: 9400 nodes, 300000 cores, 1.15PB RAM

Repository accessible thru *project-atlas.org*
More traces coming soon! You can contribute!

Overview

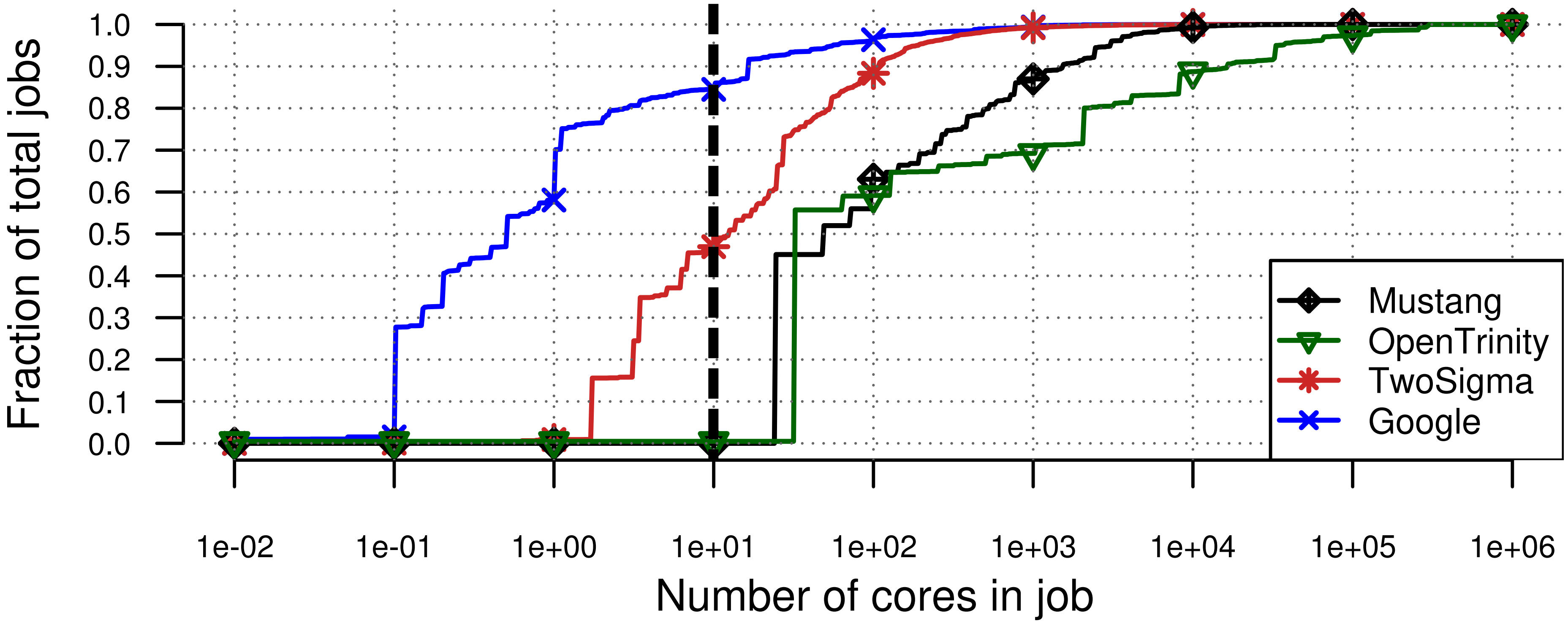
Characteristic	Google	Two Sigma	Mustang	OpenTrinity
Short jobs	Job characteristics			
Small jobs				
Diurnal patterns	Workload heterogeneity			
High job submission rate				
Resource over-commitment	Resource utilization			
Sub-second interarrival periods				
User request variability				
High failure rates	Failure analysis			
Costly failures (wasted CPU hours)				
Longer/larger jobs fail more often				

Job Sizes

- Google jobs request 3 - 406x fewer CPU cores

Two Sigma ↖ LANL ↗

- LANL request sizes more uniformly distributed

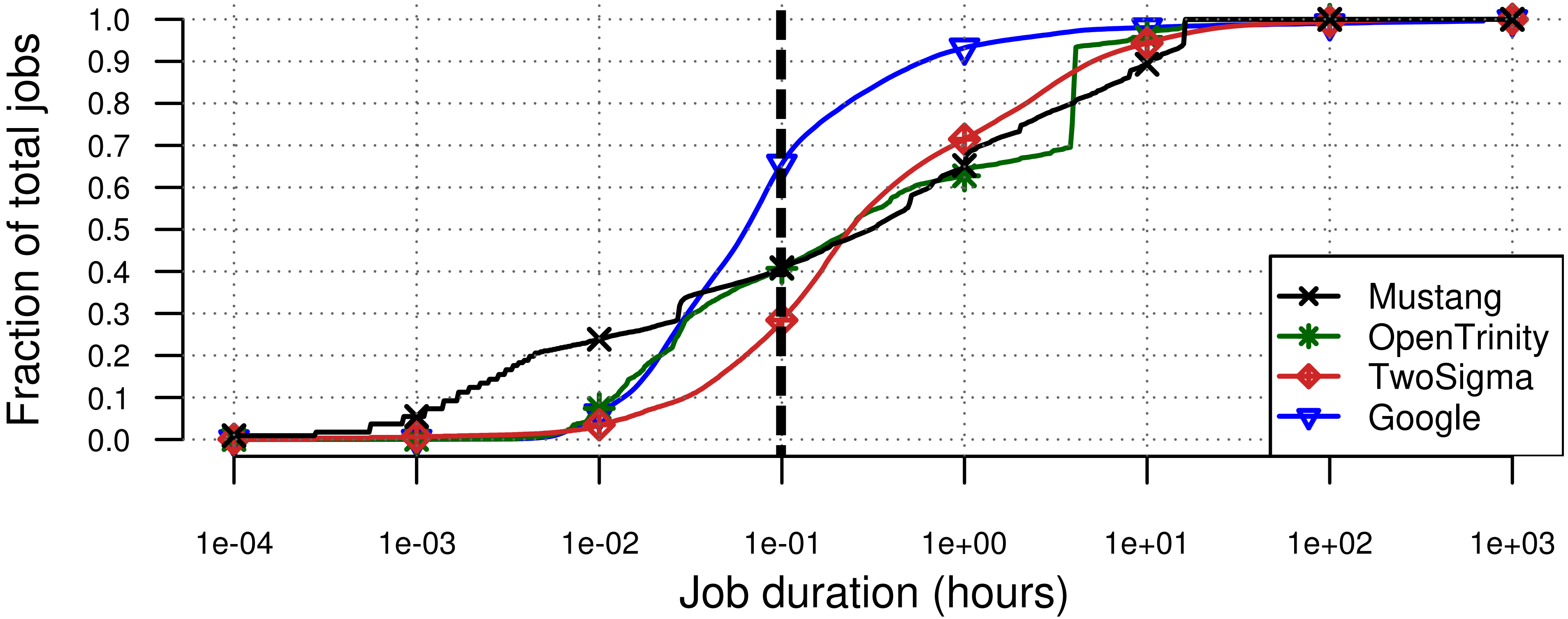


Solving head-of-line blocking by dedicating resources to small jobs becomes challenging [Delgado et al.]

Job Duration

Two Sigma LANL

- Median Google job is 4 - 5x shorter
- But: LANL jobs end at 16-32 hours, Google jobs don't



Mitigating straggler effect thru short task replication should be applied judiciously [Ananthanarayanan et al.]

Overview

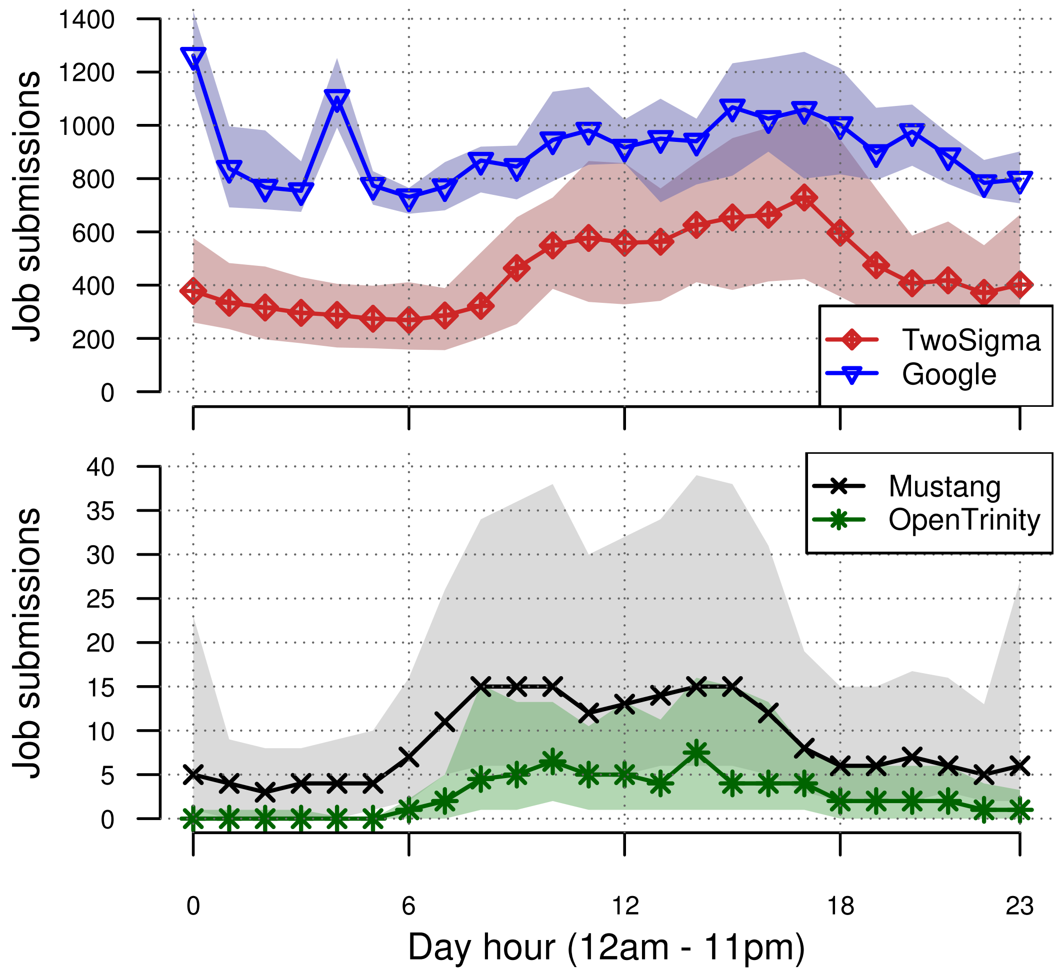
Characteristic	Google	Two Sigma	Mustang	OpenTrinity
Short jobs	✓	✗	✗	✗
Small jobs	✓	✗	✗	✗
Diurnal patterns	**Workload heterogeneity**			
High job submission rate				
Resource over-commitment	**Resource utilization**			
Sub-second interarrival periods				
User request variability				
High failure rates	**Failure analysis**			
Costly failures (wasted CPU hours)				
Longer/larger jobs fail more often				

Workload Heterogeneity

- Reversed diurnal patterns
  - More/smaller Google jobs between midnight and 4AM
- Job submission rate
  - 10-1000x more scheduling requests in Two Sigma, Google

1K jobs/hour → 3.6 sec/job

70K tasks/hour → 51 msec/task



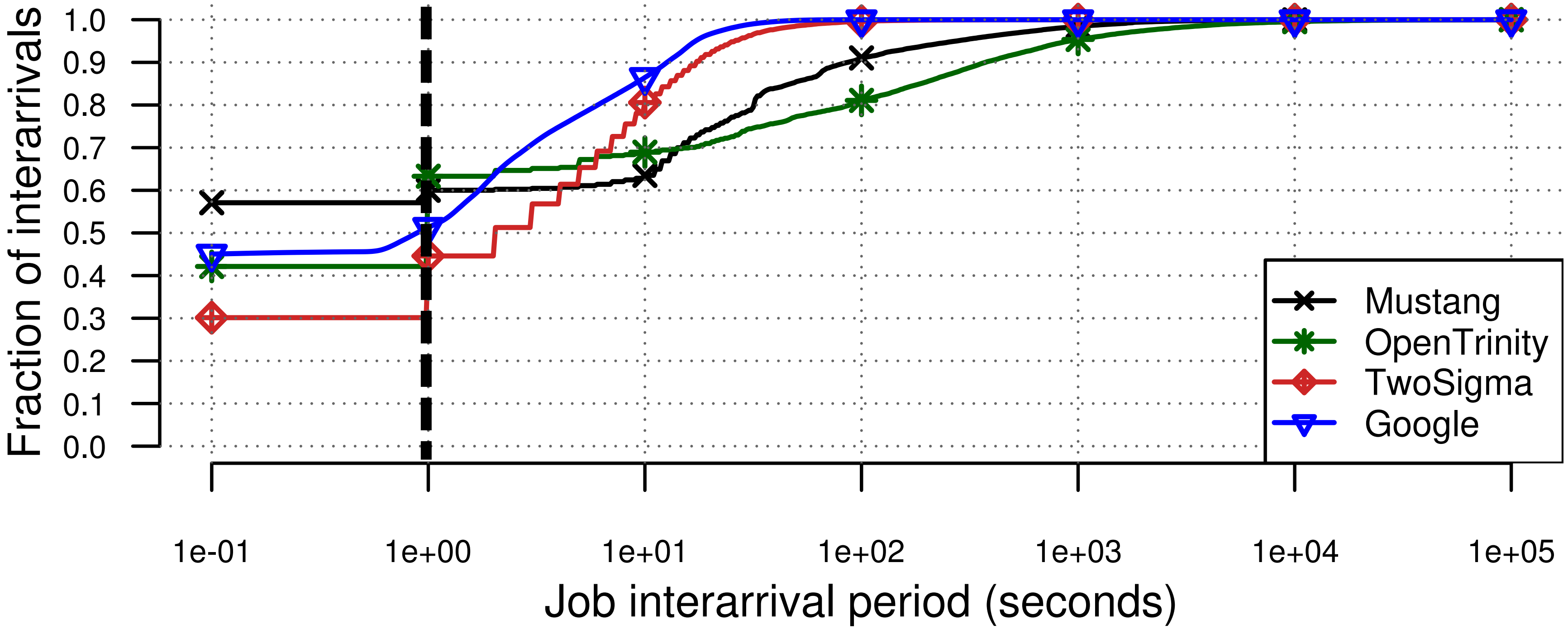
Task placement algorithms achieve subsecond latency today [Quincy, Firmament] but we should aim for msec latencies

Overview

Characteristic	Google	Two Sigma	Mustang	OpenTrinity
Short jobs	✓	✗	✗	✗
Small jobs	✓	✗	✗	✗
Diurnal patterns	✗	✓	✓	✓
High job submission rate	✓	✓	✗	✗
Resource over-commitment	Resource utilization			
Sub-second interarrival periods				
User request variability				
High failure rates	Failure analysis			
Costly failures (wasted CPU hours)				
Longer/larger jobs fail more often				

Resource utilization: intensity

- Only Google overcommits resources (others at 65-90%)
- 43-64% of inter-arrivals <1sec long
  - 20% of inter-arrivals >100sec at LANL → Maintenance



Systems should be tested with subsecond job interarrivals [Firmament, Quasar]

Overview

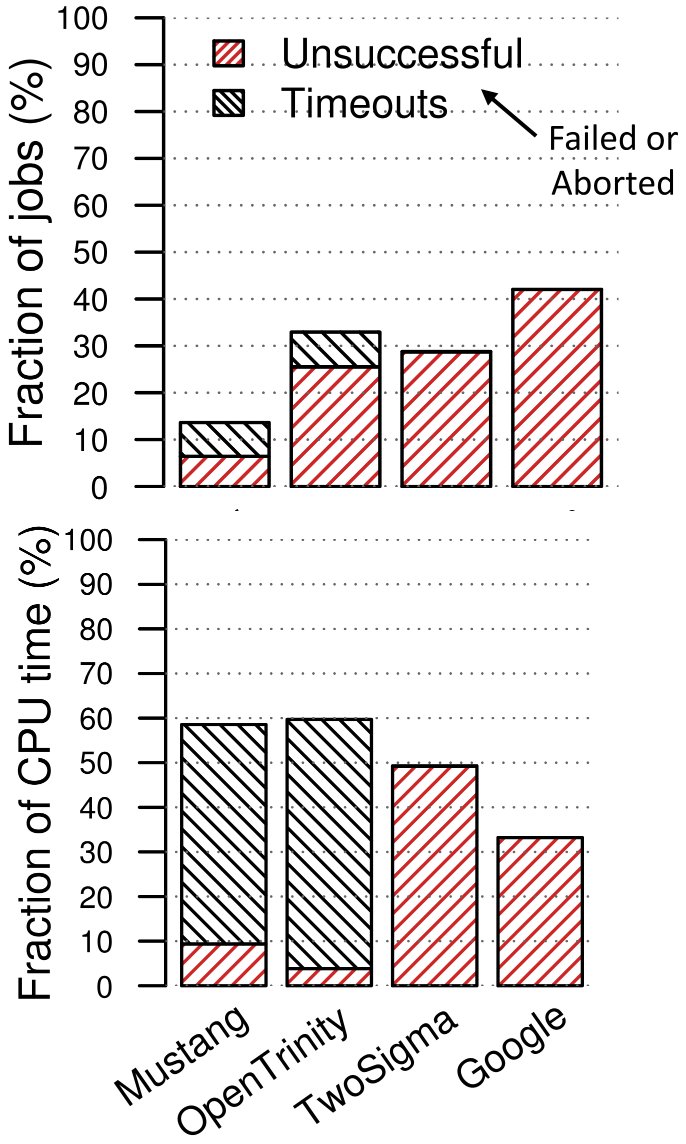
Characteristic	Google	Two Sigma	Mustang	OpenTrinity
Short jobs	✓	✗	✗	✗
Small jobs	✓	✗	✗	✗
Diurnal patterns	✗	✓	✓	✓
High job submission rate	✓	✓	✗	✗
Resource over-commitment	✓	✗	✗	✗
Sub-second interarrival periods	✓	✓	✓	✓
User request variability	✗	✓	✓	✓
High failure rates				
Costly failures (wasted CPU hours)	**Failure analysis**			
Longer/larger jobs fail more often				

Unsuccessful jobs

- Unsuccessful job rates at Google are significant
  - 1.4-6.8x higher than other traces

Two Sigma LANL

- Highest efficiency: HPC clusters
  - 34-80% fewer CPU hours wasted* at LANL
  - Time wasted <u>decreases</u> with job runtime



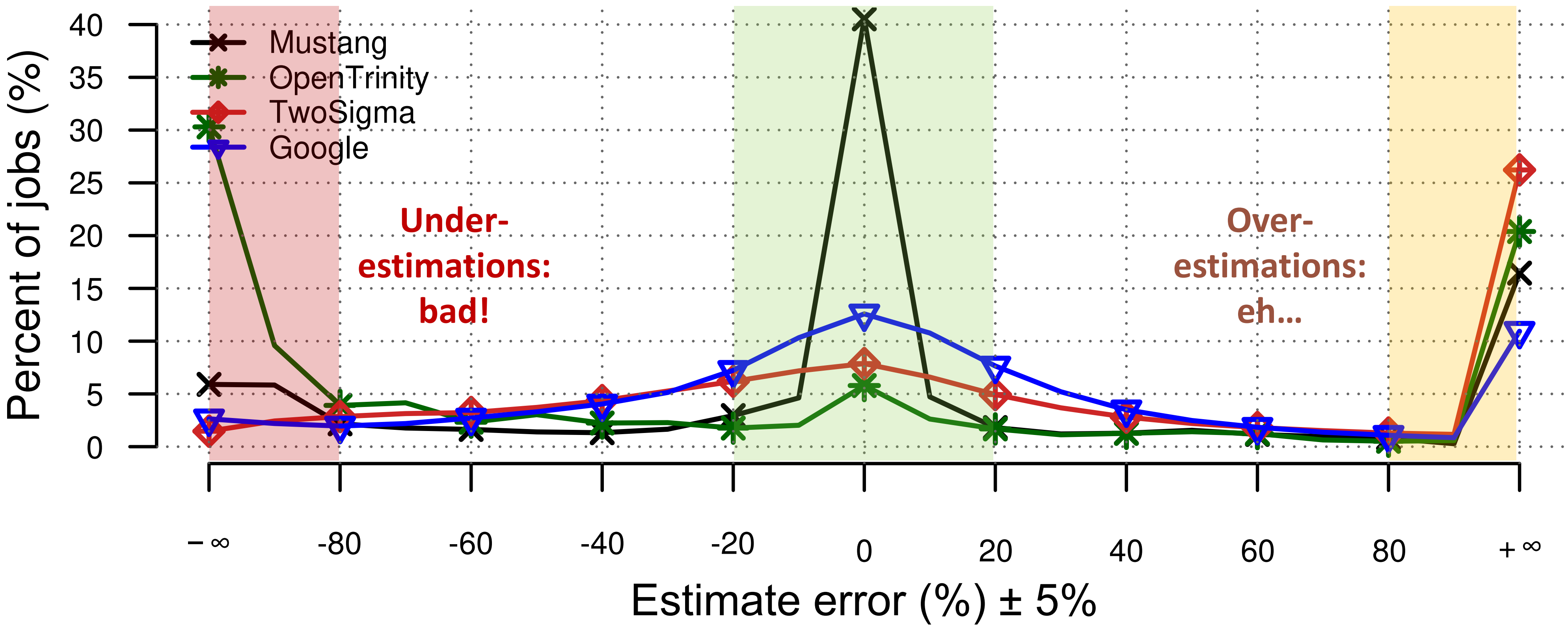
Defining *failure* is crucial: software errors may be benign

A case for dataset pluralism

Estimating job runtimes

- Runtime estimates: improve cluster efficiency
  - Adjust to heterogeneous hardware → lower response times
  - Job packing → increased utilization
- How do we come up with runtime estimates?
  - User-provided (Moab, Slurm @ LANL) → mostly inaccurate
  - Leverage job repeats (Rayon in Hadoop) → effectiveness depends on workload
- JVuPredict/3Sigma: generate estimates automatically **[EuroSys 2018]**
  - Step 1: Use past runtimes of jobs with similar *feature(s)*
  - Step 2: Select predictor with highest accuracy

JVuPredict: Accuracy across traces



- Reliance on: user ID, number of cores, job name (if present)
  - Logical job names matter!
  - Need busy (100K+ jobs) or long (3+ months) traces for training

Summary

Private more similar to HPC, except:
Failure rates, Job submission rate

Characteristic	Google	Two Sigma	Mustang	OpenTrinity
Short jobs	✓	✗	✗	✗
Small jobs	✓	✗	✗	✗
Diurnal patterns	✗	✓	✓	✓
High job submission rate	✓	✓	✗	✗
Resource over-commitment	✓	✗	✗	✗
Sub-second interarrival periods	✓	✓	✓	✓
User request variability	✗	✓	✓	✓
High failure rates	✓	✓	✗	✓
Costly failures (wasted CPU hours)	✓	✓	✗	✗
Longer/larger jobs fail more often	✓	✗	✗	✗

www.pdl.cmu.edu/ATLAS