The Mystery of the Failing Jobs: Insights from Operational Data from Two University-Wide Computing Systems

Rakesh Kumar[1], Saurabh Jha[2], Ashraf Mahgoub[1], Rajesh Kalyanam[1], Stephen L Harrell[1], Xiaohui Carol Song[1], Zbigniew Kalbarczyk[2], William T Kramer[2], Ravishankar K Iyer[2], Saurabh Bagchi[1]

1: Purdue University, 2: University of Illinois at Urbana-Champaign

Supported by NSF

Overview

- Introduction
- System and Data Details
- Job Characteristics
- Analyses
  - Job Categories Based on Exit Statuses
  - Effect of Resource Usage on Job Failures
  - Predicting Job Failures and a Better Checkpointing Method
- Open Challenges
- Conclusion

Introduction

- Job failure leads to resource wastage and user dissatisfaction
- University computing clusters are uniquely challenging:
  - Heterogeneity of jobs: Compute-Intensive, Memory-Intensive, IO-Intensive
  - Varied expertise level of the users
  - Relatively smaller size of the system administration staff
- The most comprehensive dataset publicly analyzed to date in terms of variety of data sources
  - Accounting logs, resource utilization stats, failure reports
  - System-A: Less expensive HW, 617 users, ~3M jobs
  - System-B: More expensive HW, 467 users, ~2M jobs
- New insights and old insights in new environments
  - Recommendations to reduce job failure/resource wastage for both system user and system admin
  - Build an actionable failure prediction model based on resource usages

System and Data Details

- System A
  - 580 nodes, Intel Xeon E5-2670 processors, 64 GB/node, 100 MB/s local IO BW, 23 GB/s network IO BW
- System B
  - 26,868 nodes, AMD 6276 Interlagos processors, 64 GB/node, 1.1 TB/s network IO BW
- Data
  - Accounting logs
  - Resource Utilization Stats
    - 5-minute granularity for System A and 1-minute granularity for *System B*
  - Node Failure Reports

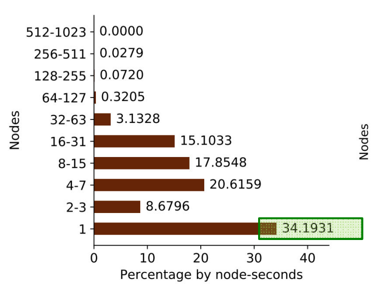
Summary of Data Analyzed

Computing Cluster		*System A*	*System B*
Duration		Mar 2015-Jun 2017	Feb-June 2017
# jobs		2,908k	2,219k
shared	# single	1,125k (38.7%, 15.8%)	-
	# multi	28k (1.0%, 1.9%)	-
	total	1,153k (39.7%, 17.7%)	-
non-shared	# single	1,348k (46.3%, 18.4%)	1,640k (73.9%, 5.4%)
	# multi	407k (14.0%, 63.9%)	580k (26.1%, 94.6 %)
	total	1,755k (60.3%, 82.3%)	2,219k (100%)
# unique users		617	467

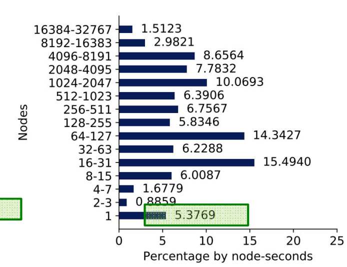
- All production jobs
- Node-seconds = #nodes x execution time
- The percentages in parenthesis refers to the raw counts and node-seconds
- Sharing allows multiple jobs to run on the same node
  - System A: 39.7% by count and 17.7% by node-seconds

Job Characteristics

- Job size
  - Single node jobs by count
    - System A: 85%, System B: 74%
  - Single node jobs by node-seconds
    - System A: 34%, System B: 5%



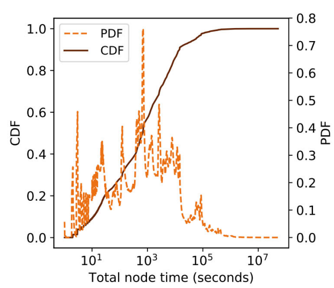
System A



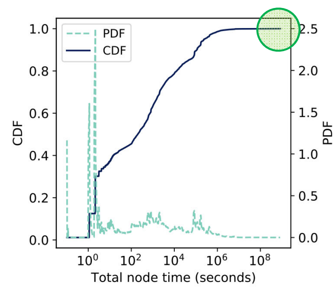
System B

Job Characteristics

- Job node-seconds
  - System A and System B: 50% of the jobs run for less than ~10^3 node-seconds
  - System B: Jobs run up to ~10^9 node-seconds



System A



System B

PURDUE UNIVERSITY

Job Categories Based on Exit Statuses

Table: Job categories based on exit codes. Percentages in brackets are based on the total node-seconds

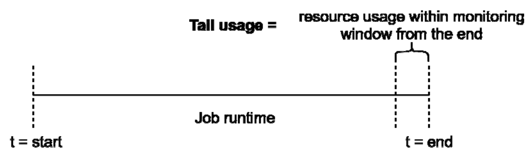
		Environment & Job Type							
		System A					System B		
		shared		non-shared		overall	non-shared		overall
		single	multi	single	multi		single	multi	
Category	Success	93.1%	87.6%	87.6%	61.8 %	86.1% (48.4%)	91.6%	64.0%	84.4% (44.4 %)
	System	2.7%	6.5%	6.5%	8.8 %	5.3% (4.0%)	0.10%	1.0%	0.3% (1.4%)
	User	1.6%	2.2%	3.5%	7.2%	3.3%(12.9%)	3.8%	3.0%	3.6% (2.7%)
	User/System	0.6%	0.2%	0.4%	6.1%	1.3% (1.3%)	1.2%	0.8%	3.6% (8.0%)
	Walltime	2.0%	3.5%	2.0%	16.1%	4.0% (33.4%)	3.7%	20.4%	8.0% (43.4%)
Total		1,125k	28k	1,348k	407k	2,908k	1,640k	579k	2,219k

- Failure categories - System, User, User/System
- System related failures - System A: 5.3%, System B: 0.3%
- Success category
  - Multi-node - System A: 61.8% (non-shared), System B: 64.0% (non-shared)
  - Single-node - System A: 93.1% (shared) vs 87.6% (non-shared), System B: 91.6%
    - Sharing does not negatively impact the jobs failure probability.
- Walltime category by node-seconds - System A: 33.4%, System B: 43.3%

PURDUE UNIVERSITY

Effect of Resource Usage on Job Failures

- Job failure rate is defined as the fraction of jobs that fail due to system related issues
- All analyses conducted using tail utilization values

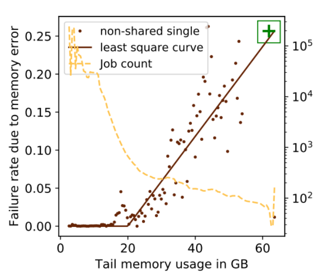


- Hypothesis testing for all correlation studies
- Resource usage prediction models based on user profiling
  - Last: same resource usage as last finished job of a given user
  - Average: average resource usage of last ‘n’ finished job of a given user
  - Median: median resource usage of last ‘n’ finished job of a given user
  - Maximum cosine similarity: same resource usage as the most similar job based on cosine similarity

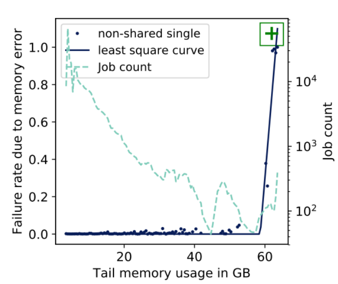
Effect of Resource Usage on Job Failures

- Memory
  - Single-node jobs: +ve correlation
  - Multi-node jobs: no correlation
    - 99th percentile value:
      - System A – 11.7 GB
      - System B – 45.6 GB

System	Job Type	Correlation coef., p-value
System A	Non-shared single	0.83, 1.7e-28
	Non-shared multi	0.17, 0.4
	Shared single	0.84, 7.2e-32
System B	Non-shared single	0.57, 3.2e-9
	Non-shared multi	0.13, 0.2



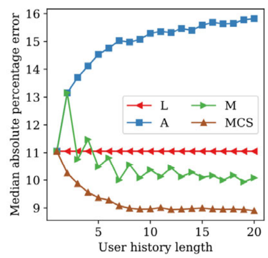
System A



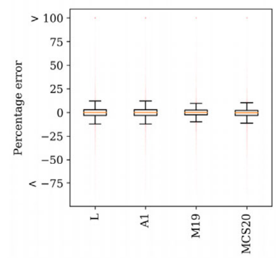
System B

Resource Usage Prediction by User Profiling

- Memory (System A)
  – MAPE of all predictors are less than 12% (for at least one history length)
  – Maximum Cosine Similarity (MCS) outperforms others
  – Use case: Predict memory usage in advance
    - Better scheduling for heterogeneous memory cluster
    - Better scheduling when sharing is enabled



(a) Different models performance with different history lengths on the training set

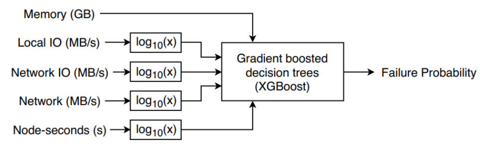


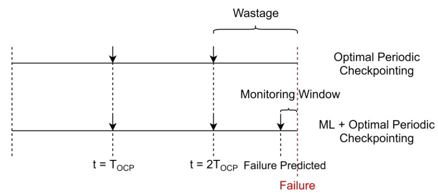
(b) Percentage error distribution for different models on test set with best history length as per training set.

Summary: Effect of Resource Usage on Job Failure

- Random IO access requests lead to failure even at ~1% of BW
  – Local IO (System A)
    - BW of 100MB/s while failure rate starts rising with utilization as low as 3 MB/s (shared) - 6MB/s (non-shared)
  – Remote IO (System B)
    - BW of 1.1TB/s while failures are observed with a utilization of only 46MB/s for a given job
- Contention at remote resources (outside node) dominant in non-shared environment, while the contention at local resources (at node) dominant in shared environment.
  – Use user-based resource usage prediction while making scheduling decisions
  – Use dynamic reconfiguration of applications based on current resource availability, such as reconfiguring the number of threads or network timeout.

Predicting Job Failure



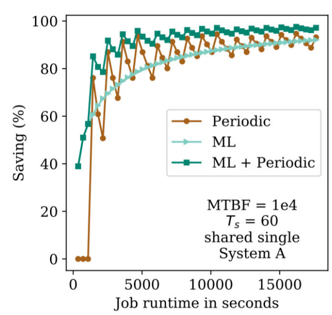


- ML model
  - Input: current resource usages, Output: failure probability within the next monitoring window
- Better checkpointing method
  - Combine our ML model with the optimal periodic checkpointing method

A Better Checkpointing System

Normalized area under the curve (normalized with respect to jobs with no wastage execution due to failures).

	System		Periodic	ML	ML+Periodic
MTBF=1e4, T_S=60 sec	A	shared single	0.81	0.82	0.91
		non-shared single		0.89	0.94
		non-shared multi		0.90	0.95
	B	non-shared multi		0.91	0.94
MTBF=1e5, T_S=60 sec	A	shared single	0.66	0.82	0.88
		non-shared single		0.89	0.92
		non-shared multi		0.90	0.93
	B	non-shared multi		0.91	0.93
MTBF=1e6, T_S=60 sec	A	shared single	0.30	0.82	0.84
		non-shared single		0.89	0.90
		non-shared multi		0.90	0.91
	B	non-shared multi		0.91	0.92
MTBF=1e6, T_S=10 sec	A	shared single	0.60	0.95	0.95
		non-shared single		0.97	0.97
		non-shared multi		0.97	0.98
	B	non-shared multi		0.97	0.98



- ML + periodic checkpointing method outperforms the base optimal checkpointing method by between 12.3% (unreliable system with MTBF =1e4, Ts=60s) and 2X (reliable system with MTBF = 1e6 and Ts=60s).
- Savings achieved by the optimal checkpointing method in case of failure decreases as a system becomes more reliable i.e., as the MTBF increases from 1*e*4 to 1e6.

Open Challenges

- Current optimal checkpointing estimation methods take only hardware reliability (such as MTBF) into account
  - This paper integrated it with job failure likelihood information
  - A better method is to consider in addition the rate of job progress
- Current contention-aware schedulers need to profile a job first to estimate job's interference and latency-sensitivity
  - Major limitation for clusters where majority of jobs are short running
  - Use user history-based resource usage predictions to profile a job profile

Conclusion

- The most comprehensive dataset publicly analyzed to date in terms of variety of data sources
- Publicly released the dataset on which the analyses are based
- Important insights into how the clusters behave and implications for how they can be managed more effectively.

Thank You!