Blind Men and the Elephant: Piecing Together Hadoop for Diagnosis

Xinghao Pan[+], Jiaqi Tan[+],

Soila Pertet[*], Rajeev Gandhi[*], Priya Narasimhan[*]

[+]DSO National Laboratories, Singapore

[*]ECE Department, Carnegie Mellon University

Carnegie Mellon
Parallel Data Laboratory

Outline

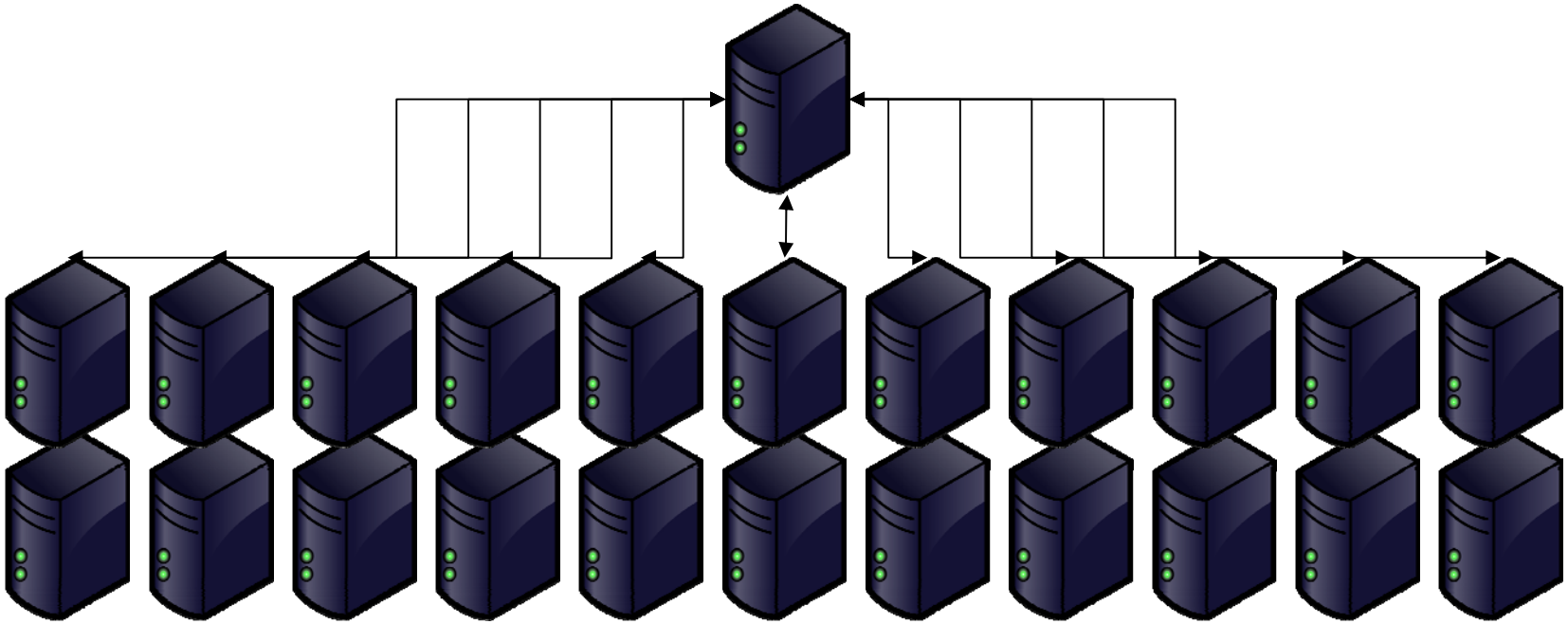
- Background
- Blimey Approach
- Instrumentation and Algorithms
- Evaluation and Results
- Conclusions

Outline

- Background
  - MapReduce and Hadoop
  - Problem Statement
- Blimey Approach
- Instrumentation and Algorithms
- Evaluation and Results
- Conclusions

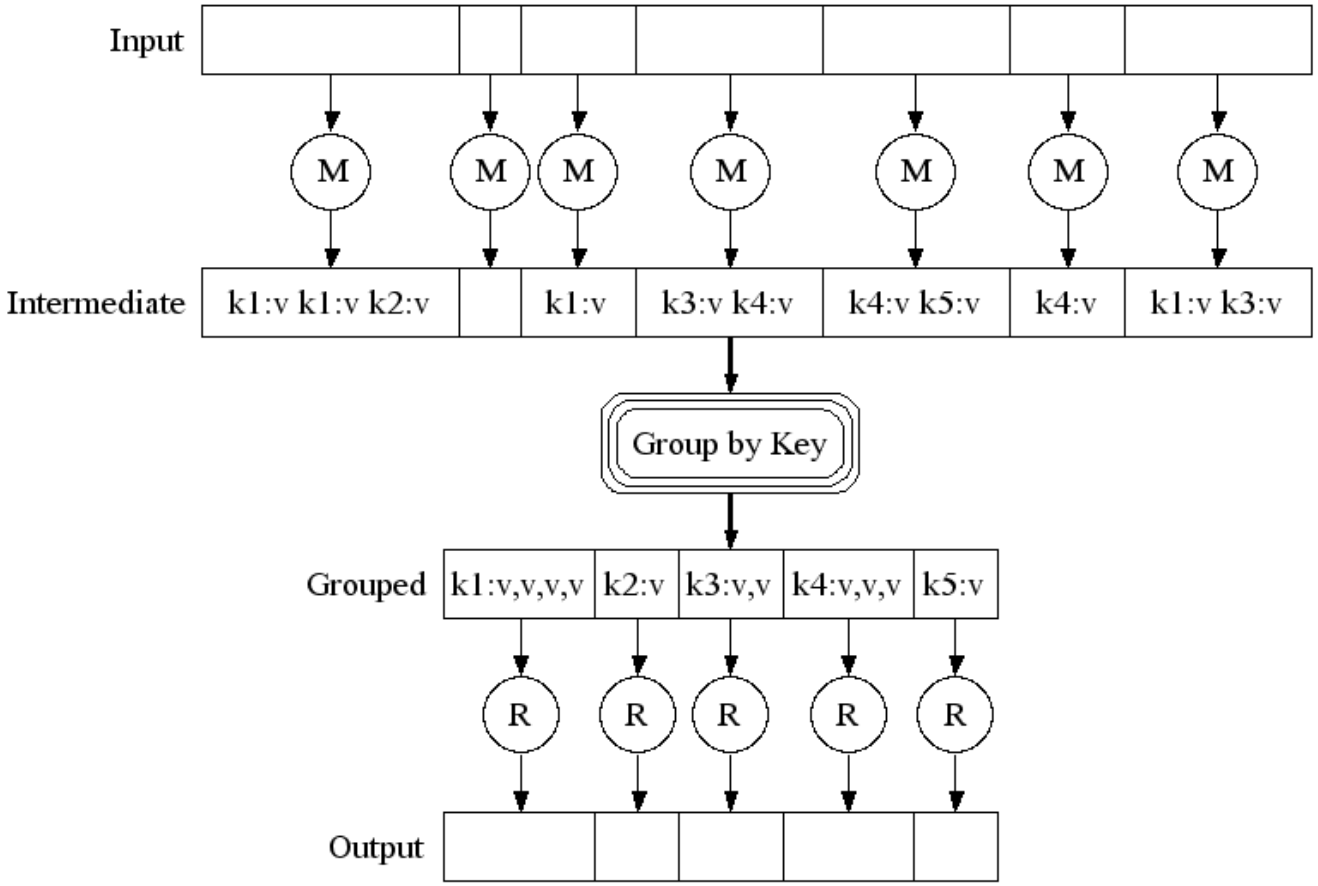
Background

- MapReduce & Hadoop
  - Increasingly popular programming framework
  - Distributed, data-intensive, parallel applications
  - Cluster of machines: single master, many slaves
  - Many Map, Reduce tasks executing same code on different input data



MapReduce Execution

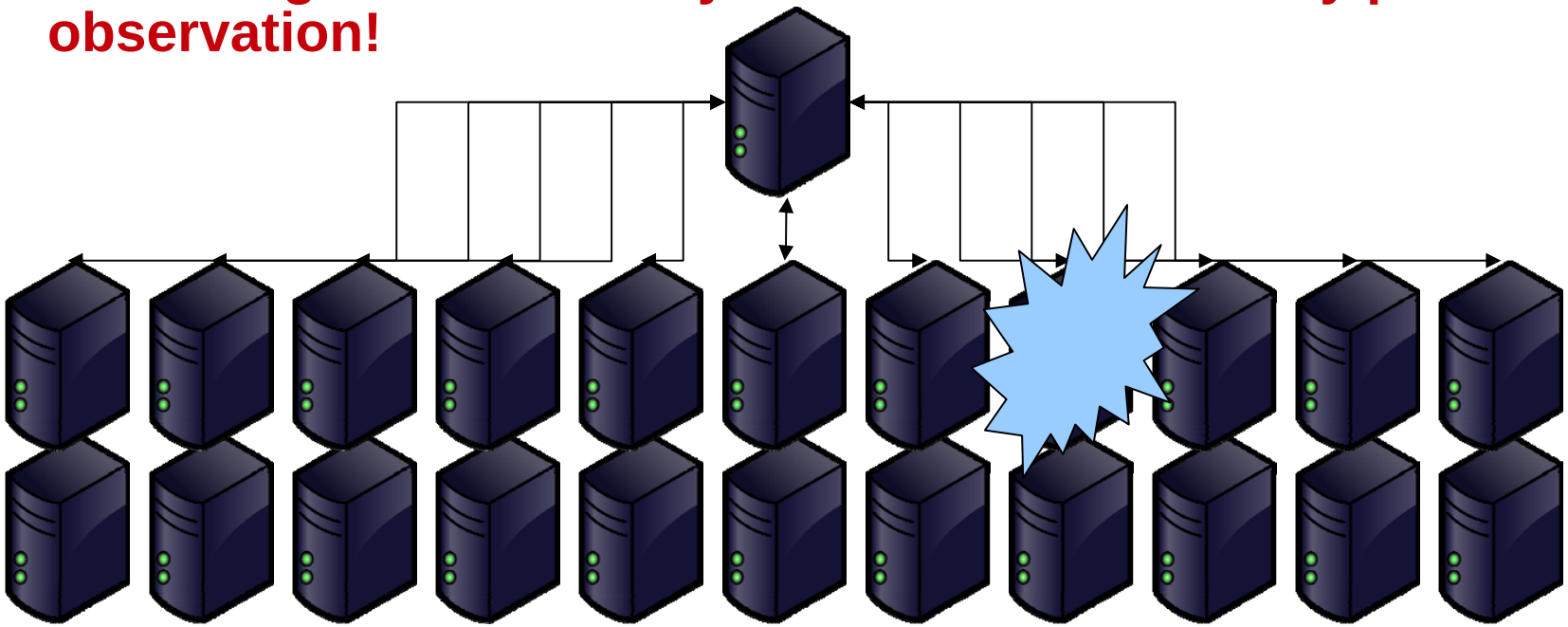
- Inputs split into blocks
- M – Map tasks; R – Reduce tasks



Source: http://labs.google.com/papers/mapreduce-osdi04.pdf

Problem

- What to do when a failure occurs?
  - Need to quickly fix faults, but..
  - Large scale – too much debugging information
  - Distributed nature – difficult for human sysadmin to reason about problem
- **IDEA: large distributed system also affords many points of observation!**



Outline

- Background
- Blimey Approach
- Instrumentation and Algorithms
- Evaluation and Results
- Conclusions

Blimey: Blind Men and Elephant

- Diagnosis framework

- Mythological story – Blind Men and the Elephant
  - Each blind man has a limited concept of the elephant
  - No blind man has complete information
  - But putting together views provide the complete picture

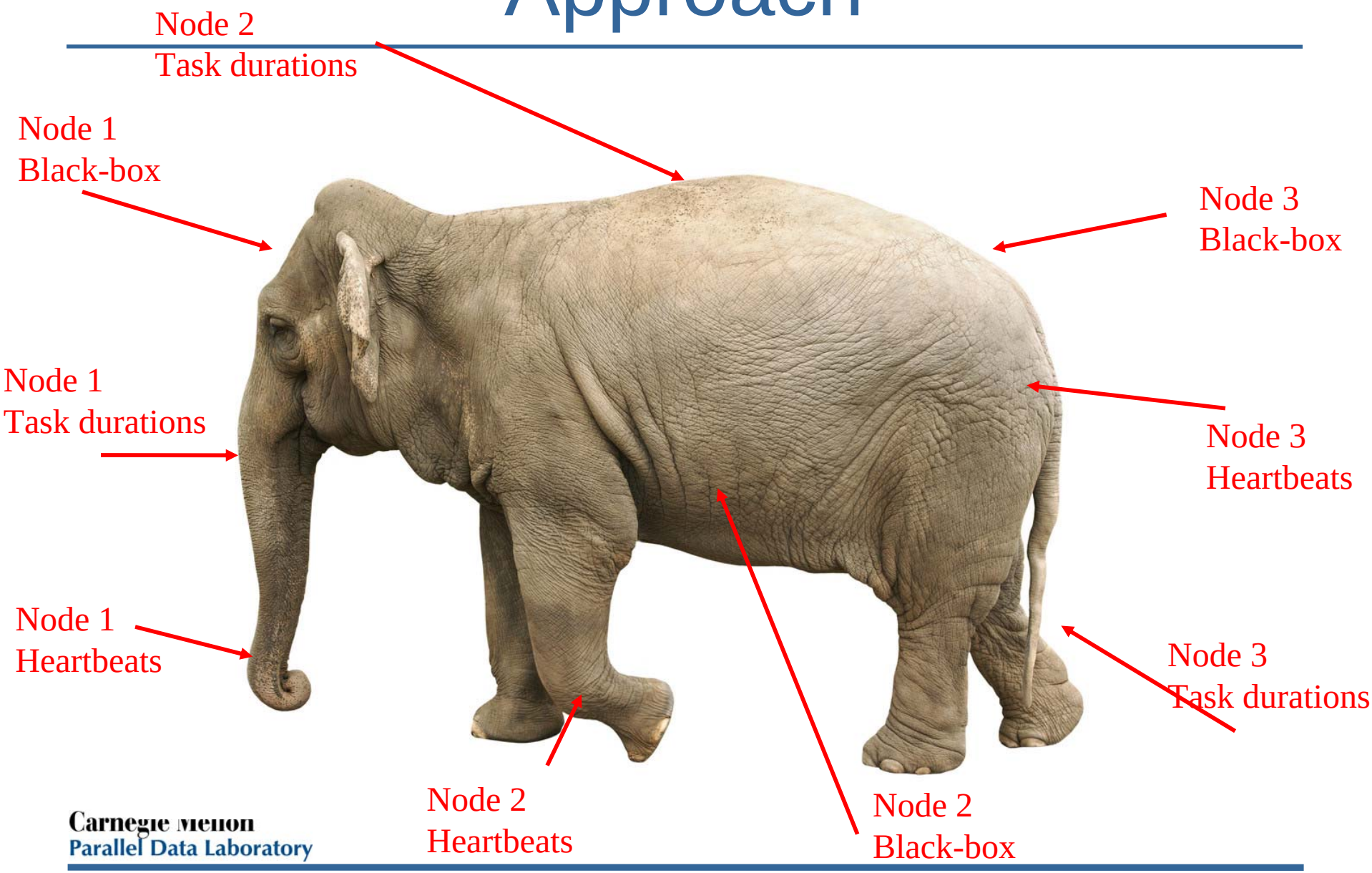
- Blimey
  - "Elephant": MapReduce system
  - "Blind men": Multiple perspectives into MapReduce system

- **Corroborate** and **synthesize** multiple distinct perspectives to *localize* and *identify* the fault

Blimey

- Blimey approach applied at two levels
- Corroborating instrumentation points
  - "Blind men": Instrumented data (e.g. OS metrics, logs) from the many cluster nodes serve as view into system
  - Diagnostic algorithms **corroborate** data to *localize* potentially faulty slave nodes
- Synthesizing outcomes from diagnostic algorithms
  - "Blind men": Diagnostic algorithms serve as secondary perspectives into system
  - We **synthesize** the secondary perspectives to *identify* the fault

Approach


Node 2
Task durations
Node 1
Black-box
Node 3
Black-box
Node 1
Task durations
Node 3
Heartbeats
Node 1
Heartbeats
Node 3
Task durations
Node 2
Heartbeats
Node 2
Black-box

Outline

- Background
- Blimey Approach
- Instrumentation and Algorithms
  - Instrumentation: Black-Box
  - Instrumentation: White-Box
  - Instrumentation: Heartbeats
  - Synthesizing Algorithms' Outcomes
- Evaluation and Results
- Conclusions

Instrumentation: Black-Box

- Black-box: external view of application
  - OS-level performance counters collected from /proc
  - Represents instantaneous behaviour of slave node
- **IDEA: slave nodes execute same code on blocks of input data; slave nodes should exhibit similar behaviors**
- *Details of diagnostic algorithms in [ISSRE09], [HotMetrics09]*

Metric	Description
user	% CPU time in user-space
system	% CPU time in kernel-space
iowait	% CPU time waiting for I/O
ctxt	Context switches per second
runq-sz	# processes waiting to run
plist-sz	Total # of processes and threads
ldavg-1	system load average for the last minute
bread	Total bytes read from disk /s
bwrtn	Total bytes written to disk /s
eth-rxbyt	Network bytes received /s
eth-txbyt	Network bytes transmitted /s
pgpgin	KBytes paged in from disk /s
pgpgout	KBytes paged out to disk /s
fault	Page faults (major+minor) /s
TCPAbortOnData	# of TCP connections aborted with data in queue
rto-max	Maximum TCP retransmission timeout

Black-Box Diagnosis

- Intuition
  - Slave nodes execute similar tasks within window of time
  - Slave nodes encountered similar workload, from black box point of view
  - Under faulty conditions, slave nodes exhibit divergent behaviour
  - Corroborate black-box metrics for diagnosis

- *Details of diagnostic algorithms in [ISSRE09], [HotMetrics09]*

Instrumentation: White-Box

- White-box (Task-centric)
  - Execution states extracted from application logs
  - Of interest: Map and Reduce durations
- **IDEA: Map/Reduce tasks execute same codes; Map/Reduce tasks should have similar durations**
- *Details of diagnostic algorithm in [ISSRE09], [WASL08]*

```
2009-04-26 22:54:25,561
   INFO org.apache.hadoop.mapred.TaskTracker:
   LaunchTaskAction: attempt_200904262253_0001_m_000000_0
.
.
2009-04-26 22:54:28,778
   INFO org.apache.hadoop.mapred.TaskTracker:
   Task attempt_200904262253_0001_m_000000_0 is done.
```

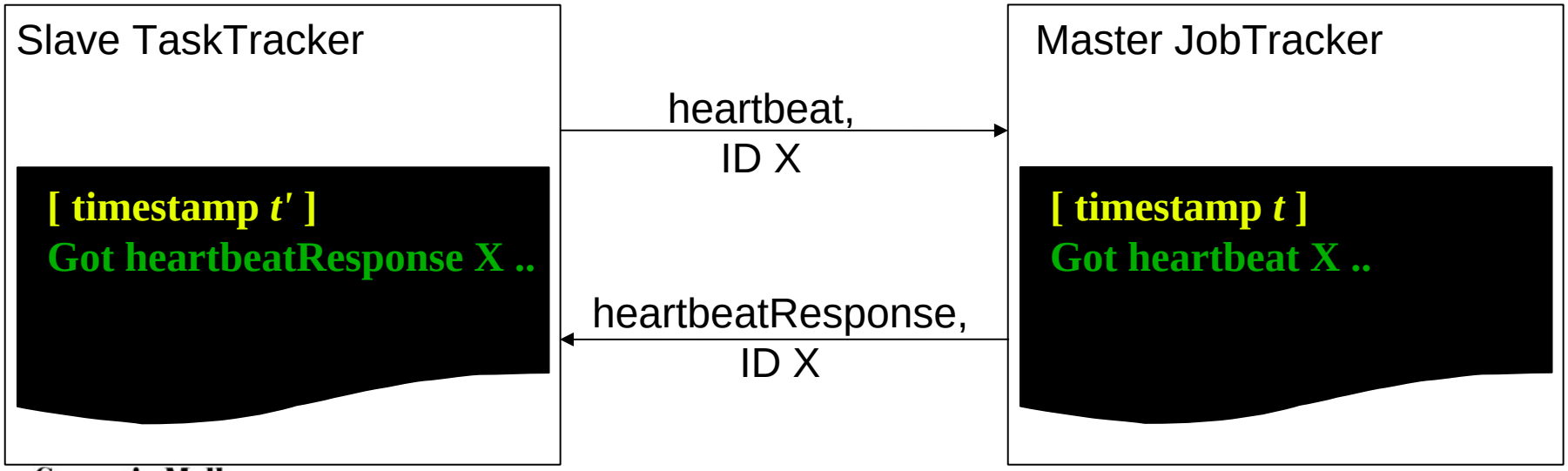
White-Box Diagnosis

- Intuition
  - Similarity of Map/Reduce tasks
  - Slave node execute subset of global set of tasks
  - Under faulty conditions, tasks on different slaves have different durations
  - Corroborate durations (as property of tasks) across slave nodes

- *Details of diagnostic algorithm in [ISSRE09], [WASL08]*

Instrumentation: Heartbeats

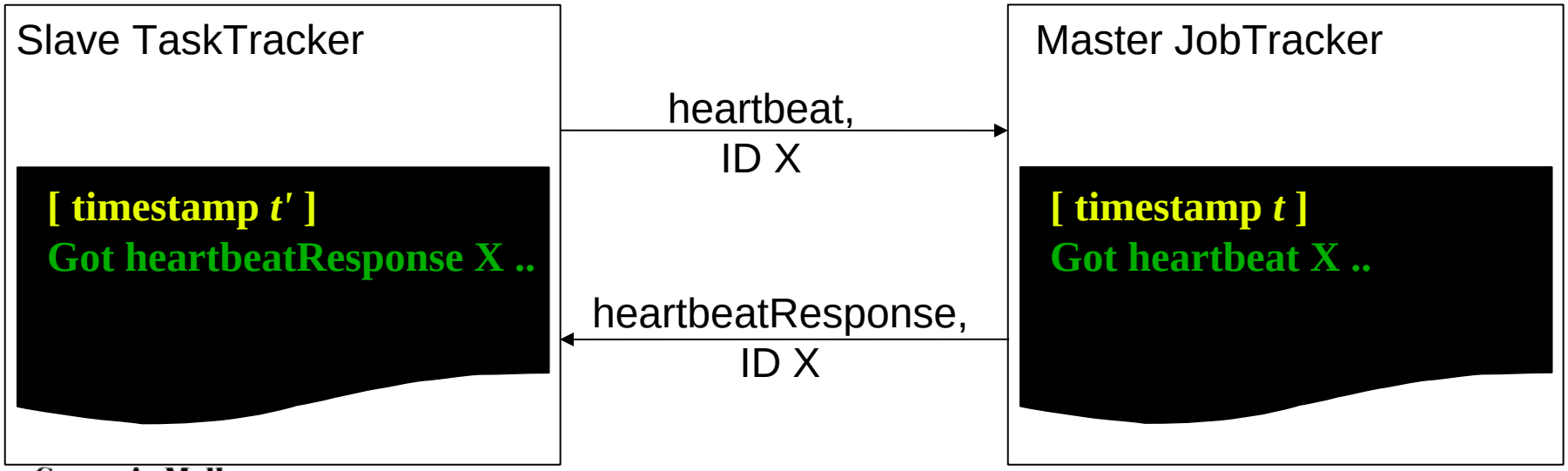
- Heartbeats
  - Keep-alive messages initiated by slave TaskTracker to master JobTracker
  - Sent periodically and upon task completion



Instrumentation: Heartbeats

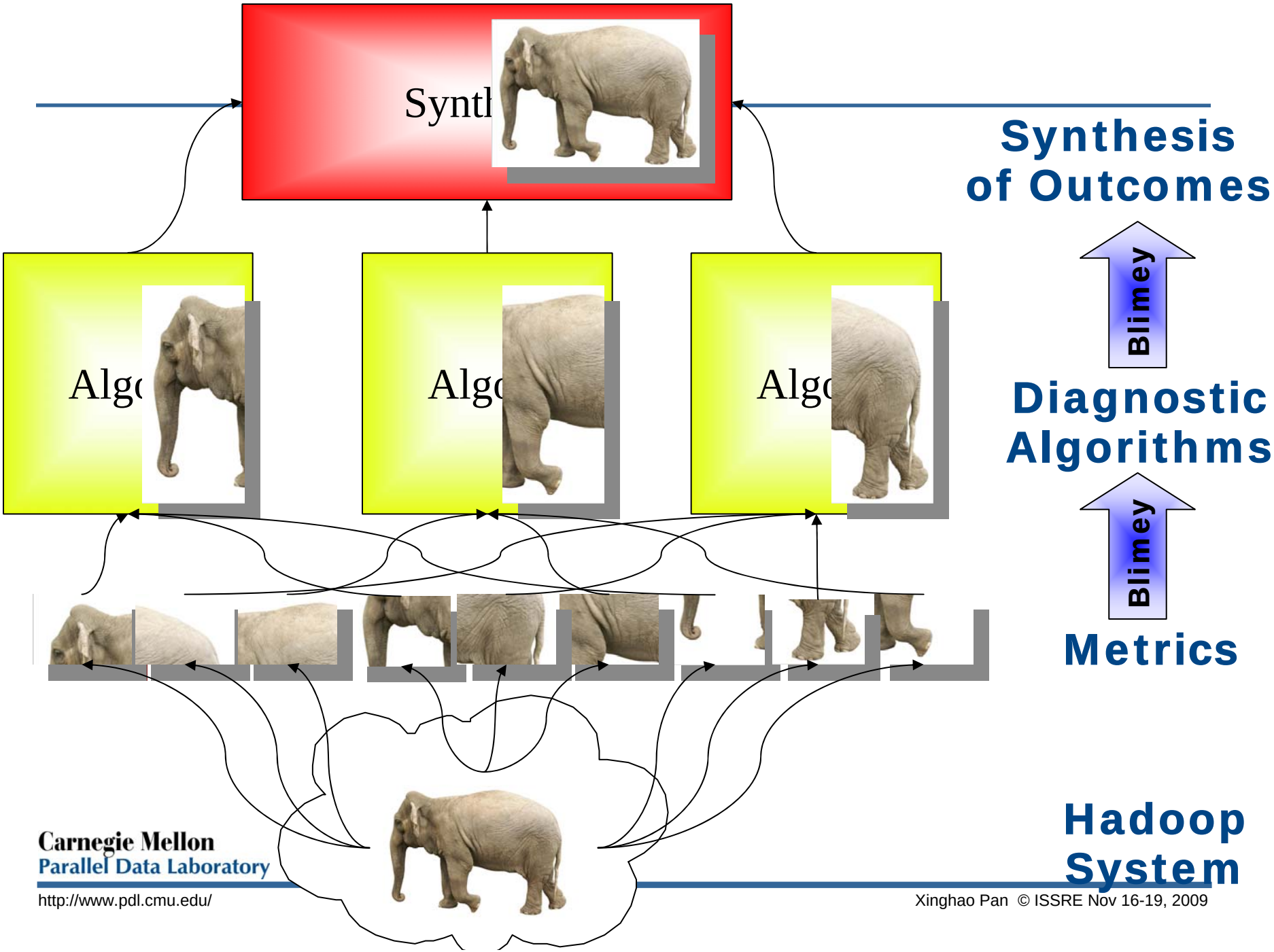
- **IDEA 1: Rate of heartbeats is a coarse indicator for workload; slave nodes should have similar workloads and heartbeat rates**
- **IDEA 2: Timestamp difference between log messages should be constant**

- *Details of diagnostic algorithms in [ISSRE09]*



Synthesizing Algorithms' Outcomes

- Different faults manifest differently on different metrics (e.g. CPU utilization, Map task duration, heartbeat rates, ...)
  - Fault may or may not manifest on particular metrics
  - Fault may or may not manifest in correlated fashion on particular metrics
- Different diagnostic algorithms operate on different metrics
- **IDEA: Synthesize algorithms' outcomes as secondary perspectives into Hadoop system**
  - Build decision tree to classify faults using outcomes from diagnostic algorithms
- *Details of diagnostic algorithms in [ISSRE09]*


Synthesis of Outcomes
Blimey
Diagnostic Algorithms
Blimey
Metrics
Hadoop System

Outline

- Background
- Blimey Approach
- Instrumentation and Algorithms
- Evaluation and Results
- Conclusions

Evaluation: Testbed & Workload

- Inject faults into Hadoop cluster
  - Fault injected on single node in cluster
  - Test accuracy of localizing and identifying faults

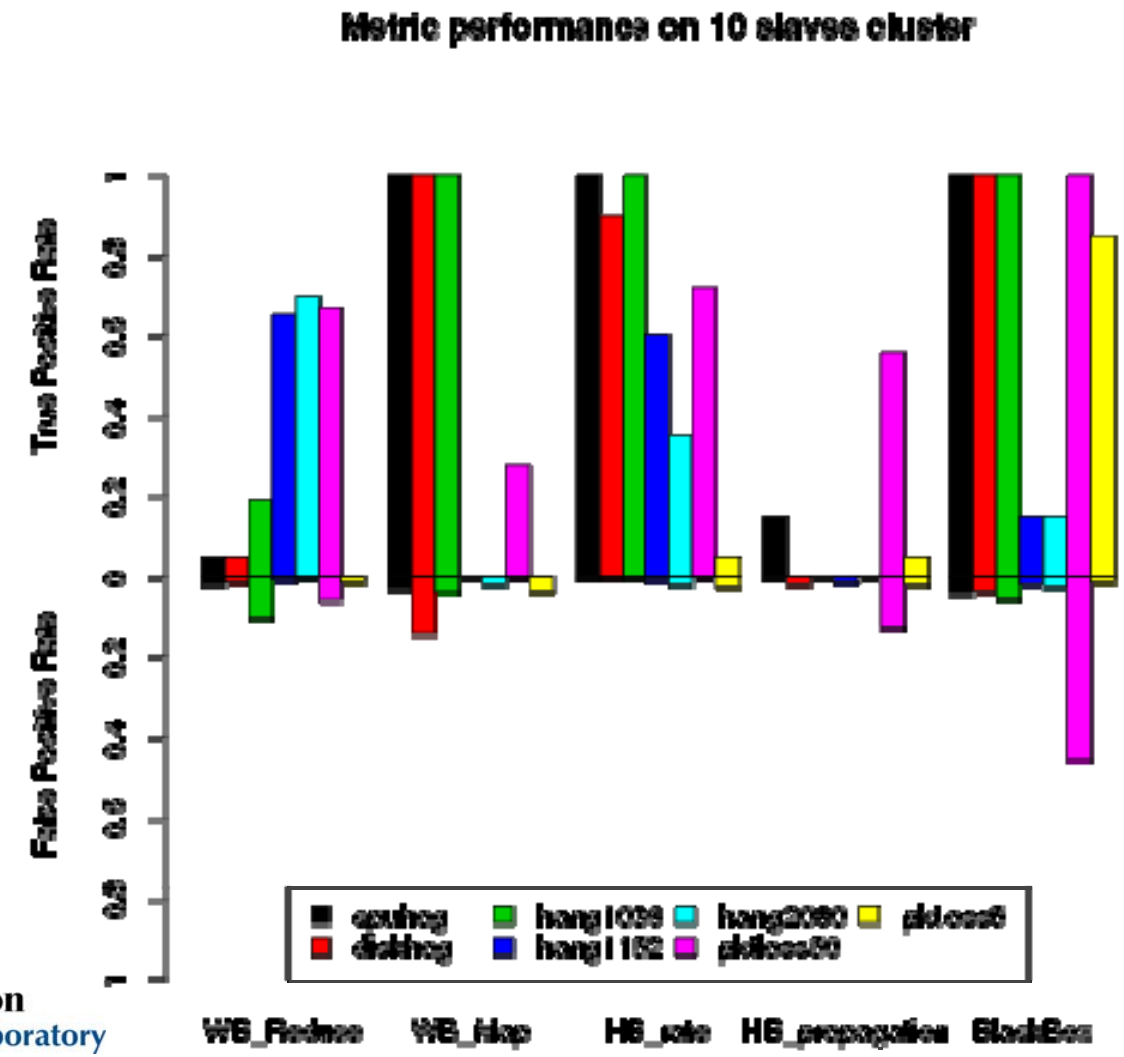
- Testbed
  - Hadoop 0.18.3
  - 10- and 50-slave nodes clusters
  - Amazon's EC2, Large instances
    - 7.5 GB RAM, two dual-core CPUs, amd64 Debian/GNU Linux 4.0

- Gridmix
  - Well-accepted, multi-workload Hadoop benchmark
  - Mimics observed data-access patterns in actual user jobs in enterprise deployments

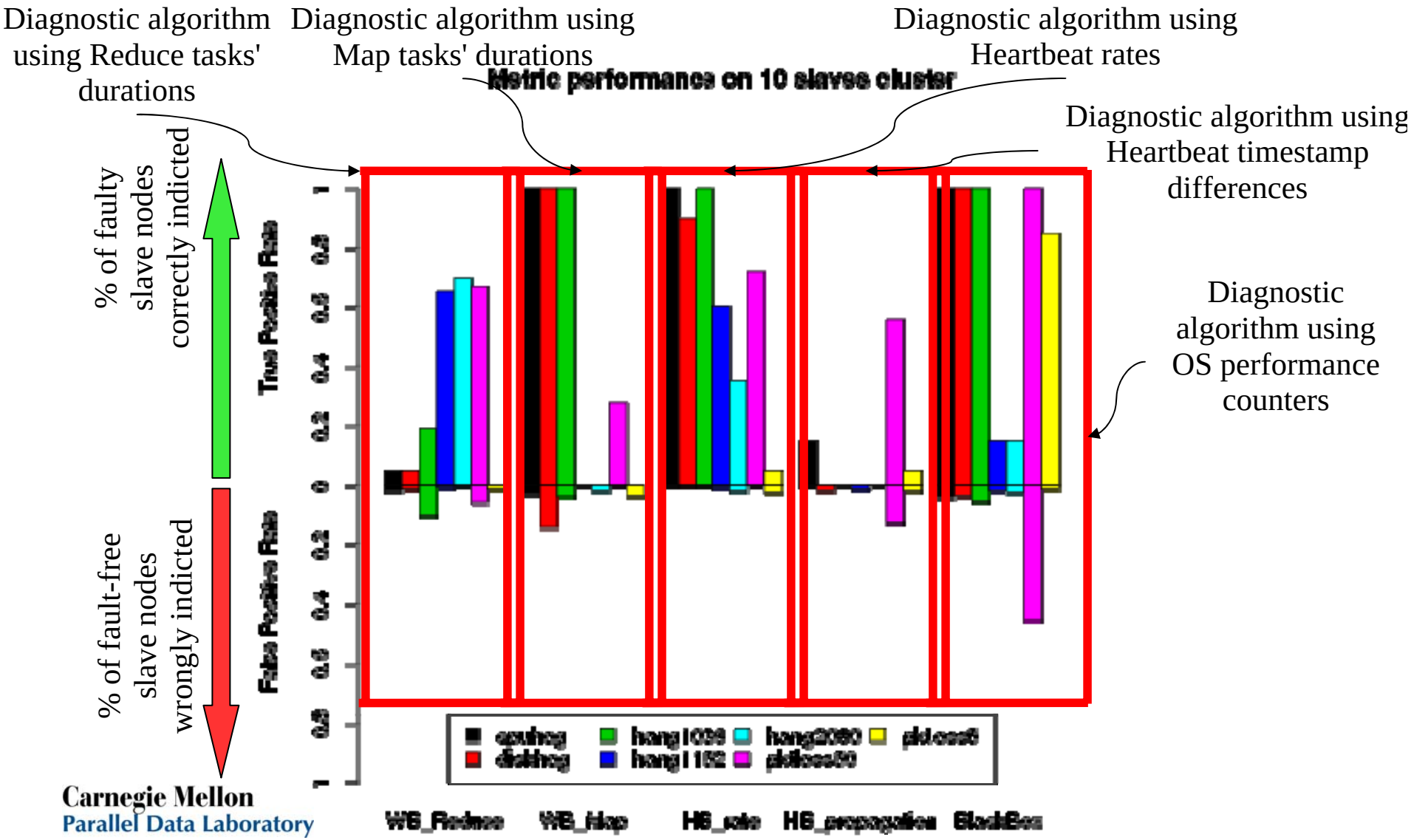
Evaluation: Fault Injection

	Fault	Description
Fault free	Control	Control experiments with no injected faults
Resource contention	CPU hog	External process uses 70% of CPU
	Packet-loss	Drop 5% or 50% of incoming packets
	Disk hog	Repeatedly write 20GB file
Application bugs Source: Hadoop JIRA	HADOOP-1036	Maps hang due to unhandled exception
	HADOOP-1152	Reduces fail while copying map output
	HADOOP-2080	Reduces fail due to incorrect checksum
	HADOOP-2051	Jobs hang due to unhandled exception

Results: Diagnosis Algorithms



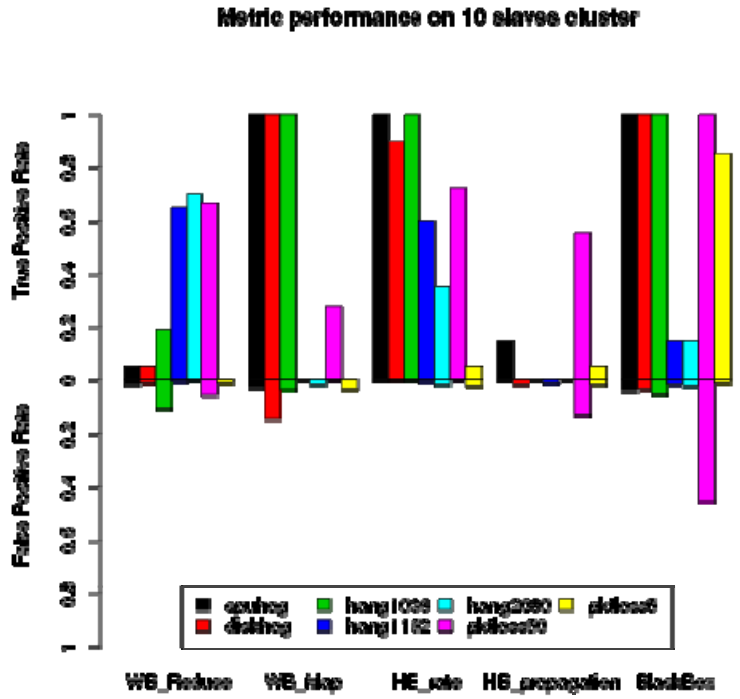
Results: Diagnosis Algorithms



Carnegie Mellon
Parallel Data Laboratory

Results: Diagnosis Algorithms

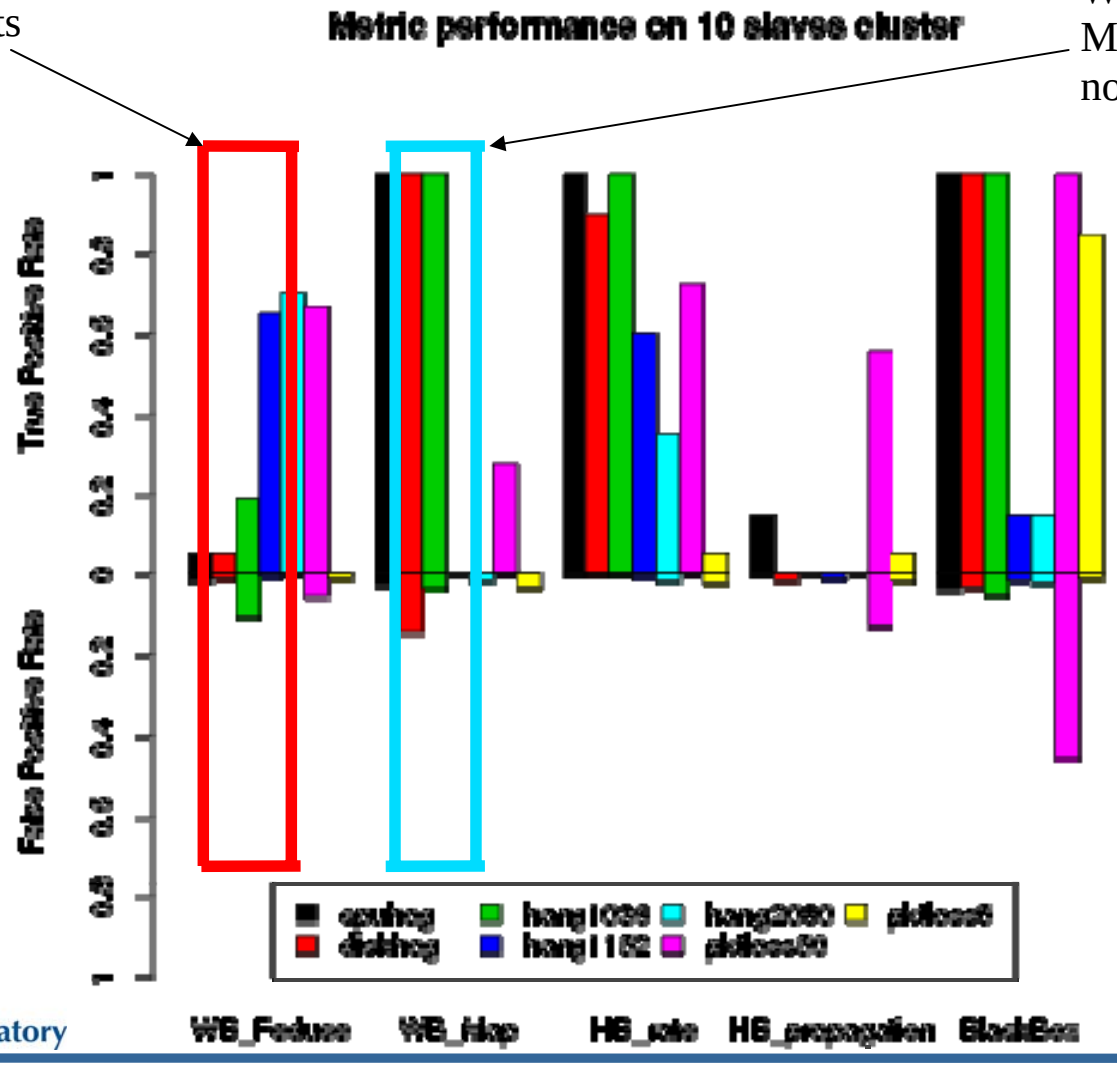
- Every fault is accurately localized by some algorithm
  - High true positive rate
  - Low false positive rate
- No single algorithm is perfect at localizing all faults
- Faults manifest differently
  - May or may not manifest
  - Varying degrees of correlated manifestation



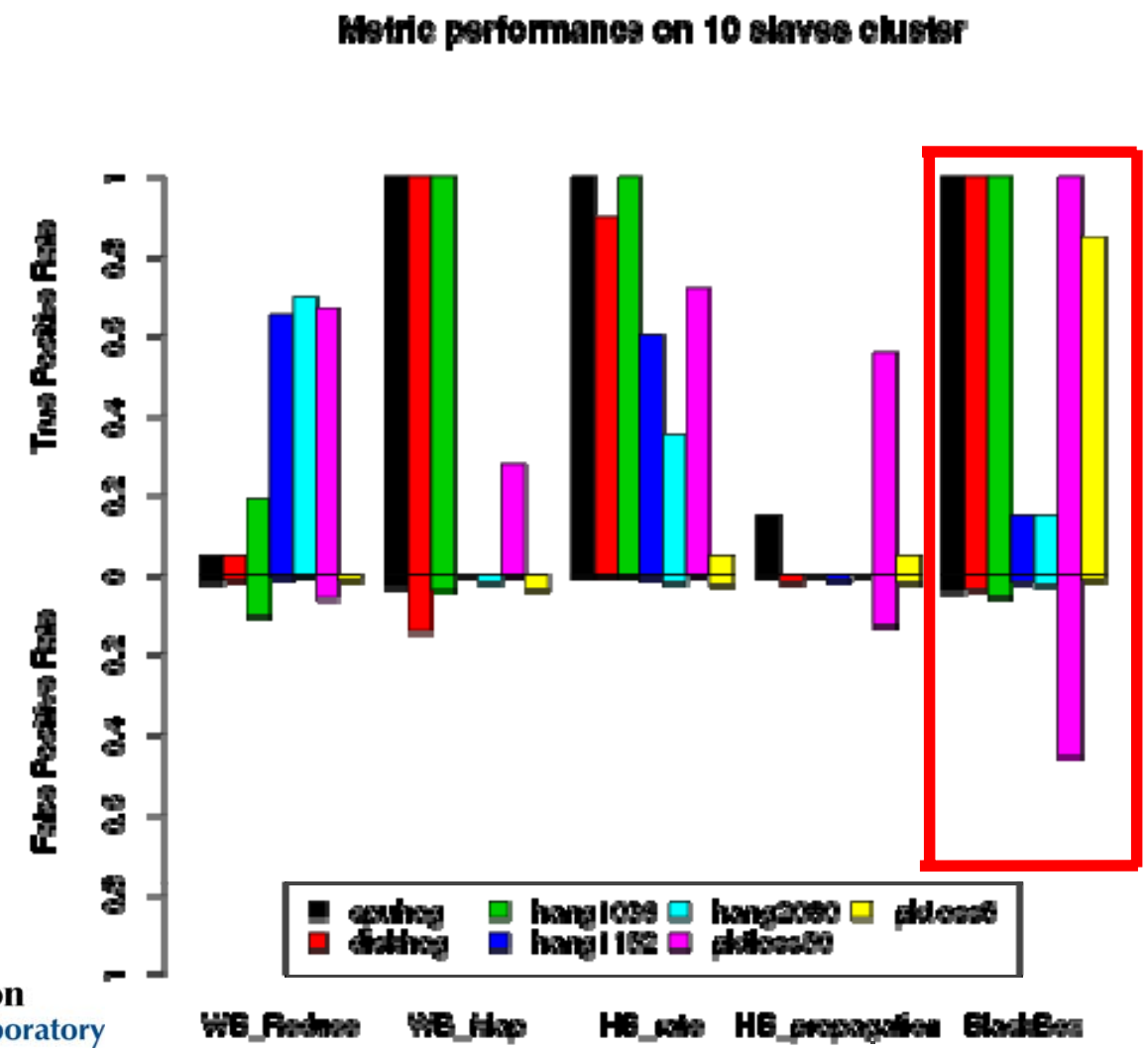
Results: Diagnosis Algorithms

WB_Reduce detects Reduce hangs but not Map hang

WB_Map detects Map hang but not Reduce hangs



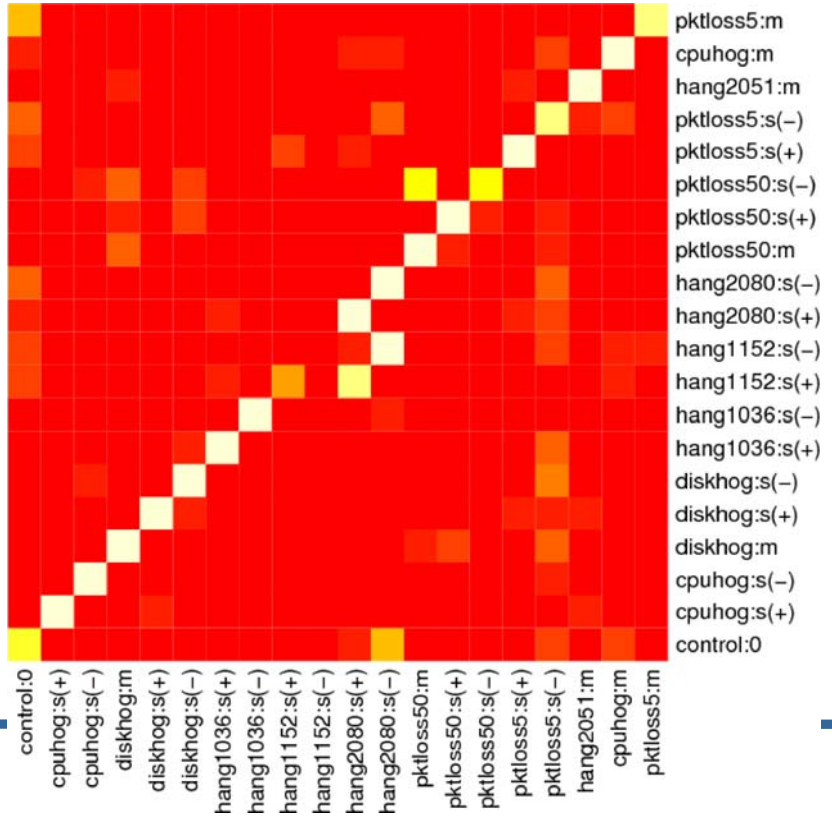
Results: Diagnosis Algorithms



Black-box detects resource-related faults

Results: Synthesis

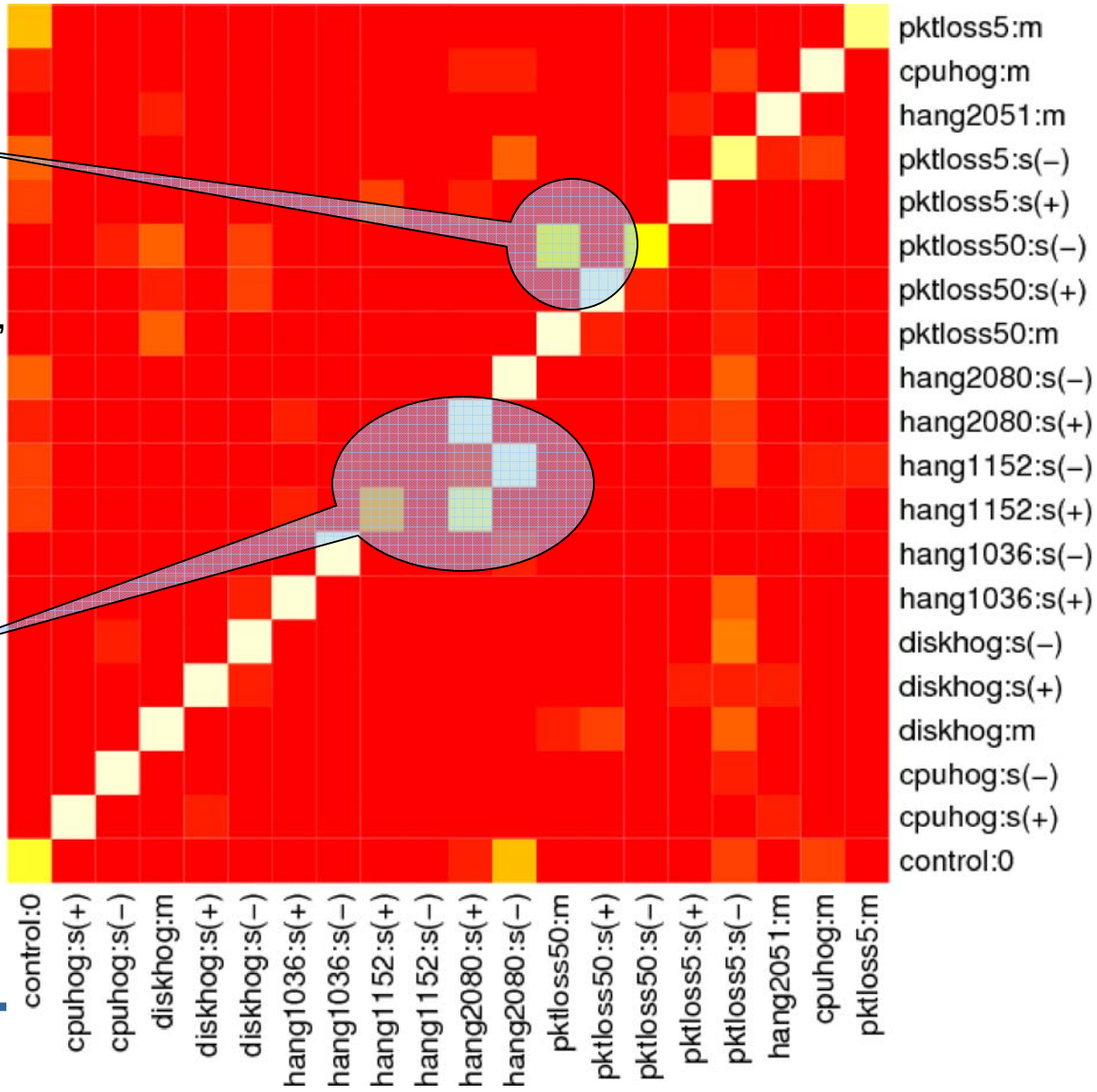
- Confusion matrix: accuracy of fault classification
  - Rows represent true class
  - Columns represent label generated by decision tree
  - Light shade means true fault likely to be given particular label
  - Ideal: light diagonals, dark everywhere else



Results: Synthesis

Fault-free slave node, affected by 50% packet loss on **another slave node**, is confused as fault-free slave node, affected by 50% packet loss on the **master node**.
Both are massive packet losses, but not at the node itself.

hang1152 as hang2080
Both are Reduce hangs; location not confused.



Outline

- Background
- Blimey Approach
- Instrumentation and Algorithms
- Evaluation and Results
- Conclusions
  - Future Work
  - Summary

Future Work

- Other instrumentation sources
- Visualization of diagnosis outcomes and data
- Fine-grain diagnosis: **why** and **how**
  - Root-cause analysis

Summary / Conclusion

- Identified multiple instrumentation points in Hadoop that can be corroborated for diagnosis

- Blimey provides a framework for failure diagnosis in MapReduce systems
  - Diagnostic algorithms corroborate multiple viewpoints to indict faulty slave nodes
  - Synthesizing secondary perspectives from algorithms, we differentiate and identify faults

Related Material

- [ISSRE09] X.Pan, J. Tan, S. Kavulya, R. Gandhi, and P. Narasimhan. Blind Men and the Elephant: Piecing Together Hadoop for Diagnosis, in 20th International Symposium on Software Reliability Engineering (ISSRE), Nov 2009.

- [HotMetrics09] X. Pan, J. Tan, S. Kavulya, R. Gandhi, and P. Narasimhan. Ganesha: Black-Box Diagnosis of MapReduce Systems, in Second Workshop on Hot Topics in Measurement and Modeling of Computer Systems, June 2009.

- **[CMU-CS-09-135] X. Pan. Blind Men and the Elephant: Piecing Together Hadoop for Diagnosis. Master's Thesis, Carnegie Mellon University, 2009. Technical Report: CMU-CS-09-135. May 2009.**

- [WASL08] J. Tan, X. Pan, S. Kavulya, R. Gandhi, P. Narasimhan. SALSA: Analyzing Logs as StAte Machines. First USENIX Workshop on Analysis of System Logs (WASL), San Diego, CA, Dec 2008.

Extra Slides

Target System: MapReduce and Hadoop

- MapReduce
  - Framework for distributed, parallel programming
  - Job = multiple copies of maps and reduces: Executed on different segments of large dataset
- Hadoop: open-source implementation
  - Distributed Runtime + Distributed FileSystem
  - Master/Slave: one Master, multiple Slaves