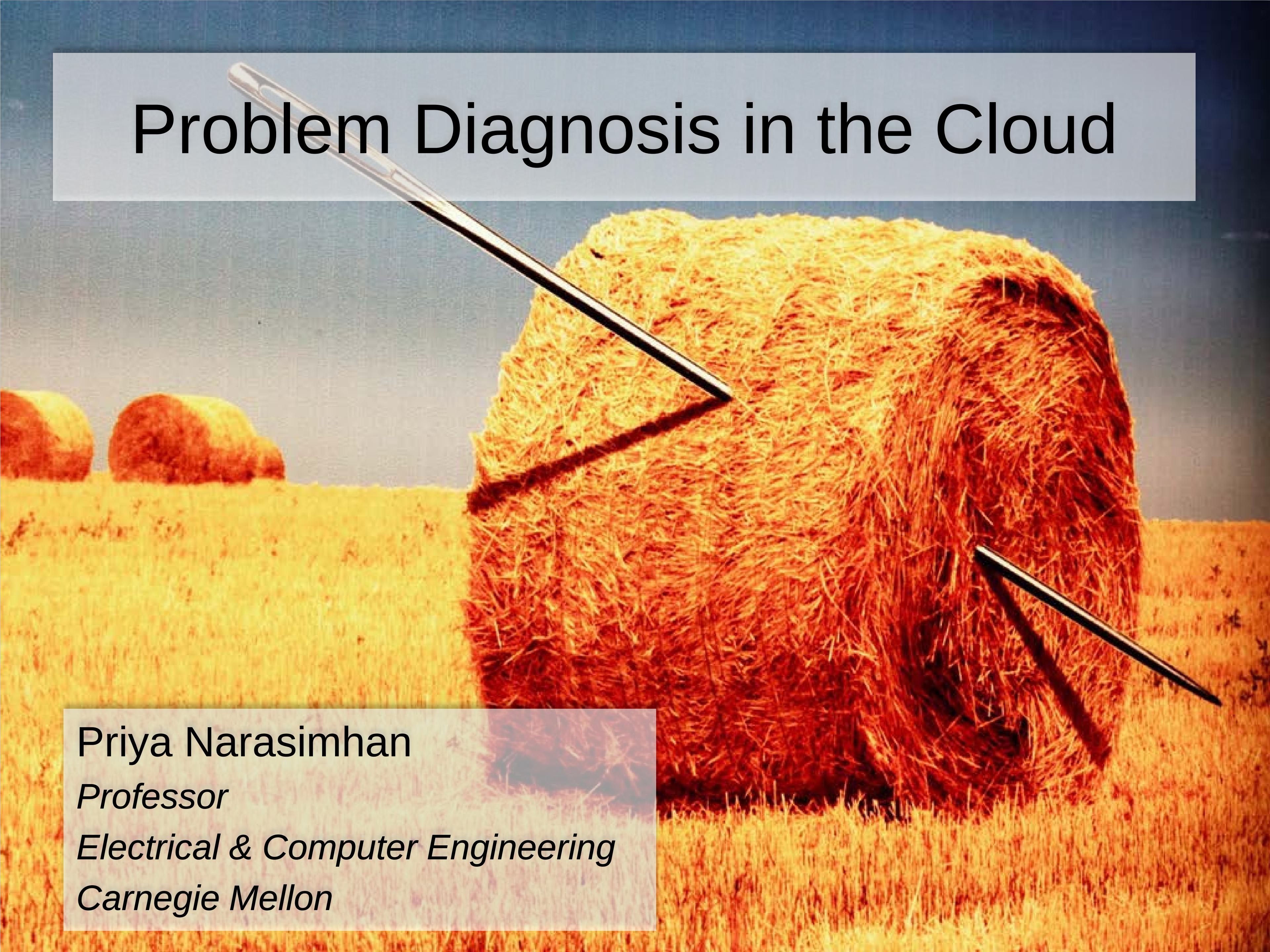


Problem Diagnosis in the Cloud

Priya Narasimhan
Professor
Electrical & Computer Engineering
Carnegie Mellon

Motivation

- Diagnosing problems
  - Creates major headaches for administrators
  - Worsens as scale and system complexity grows

- Goal: automate it and get proactive
  - Failure detection and prediction
  - Problem determination ("automated fingerpointing")
  - Problem visualization

- How: Instrumentation plus statistical analysis

Explorations

- Current explorations
  - *Hadoop*
    - [HotCloud 09, HotMetrics 09, WASL 08, SysML 08, NOMS 10, ISSRE 09, CCGrid 10, ICDCS 10, USENIX LISA 12, ICAC 13]
  - *PVFS*
    - High-performance file system (Argonne National Labs) [FAST 10]
  - *Lustre*
    - High-performance file system (Sun Microsystems) [FAST 10]

- Studied
  - Various types of problems
  - Various kinds of instrumentation
  - Various kinds of data-analysis techniques
  - Various kinds of visualization

Goals & Non-Goals

- Diagnose faulty Master/Slave node to user/admin
- Target production environment
  - Don't instrument Hadoop or applications additionally
  - Use Hadoop logs as-is (*white-box strategy*)
  - Use OS-level metrics (*black-box strategy*)
- Work for various workloads and under workload changes
- Support online and offline diagnosis
- Enable visualization of job progress for root-cause analysis

- Non-goals (for now)
  - Tracing problem down to offending line of code
  - Diagnosis of value faults

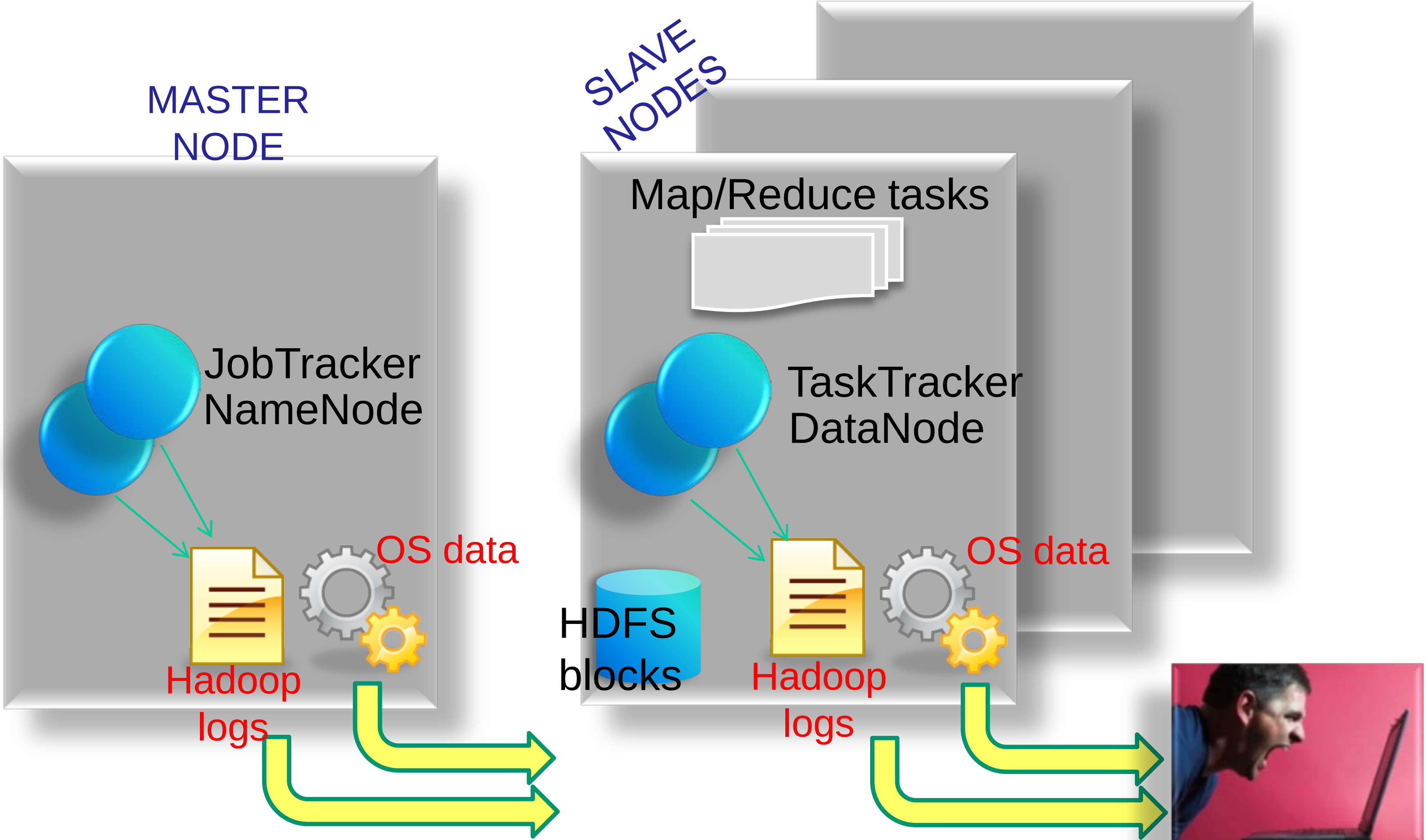
Target Hadoop Clusters

- Yahoo!'s M45 cluster
  - Production environment (managed by Yahoo!)
  - Offered to CMU as free cloud-computing resource
  - Diverse kinds of real workloads, problems in the wild
    - Massive machine-learning, language/machine-translation
  - Permission to harvest all logs and OS data each week
- Amazon's EC2 cluster
  - Production environment (managed by Amazon)
  - Commercial, pay-as-you-use cloud-computing resource
  - Workloads under our control, problems injected by us
    - gridmix, nutch, pig, sort, randwriter
  - Can harvest logs and OS data of only our workloads

Performance Problems Studied

	Fault	Description
Resource contention	CPU hog	External process uses 70% of CPU
	Packet-loss	5% or 50% of incoming packets dropped
	Disk hog	20GB file repeatedly written to
	Disk full	Disk full
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
	HADOOP-1255	Infinite loop at Nameode

Hadoop: Instrumentation



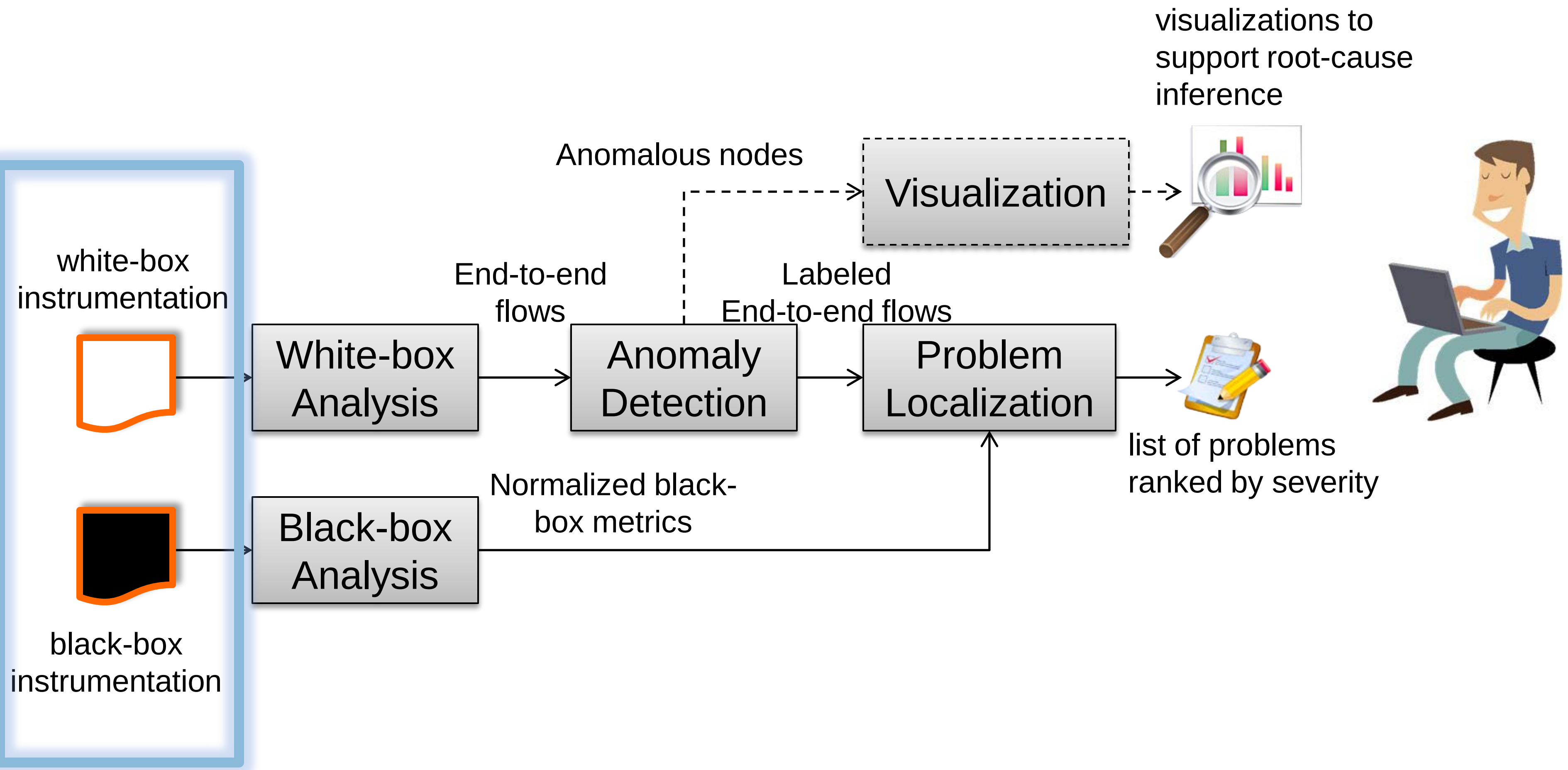
Intuition for Diagnosis

- One initial algorithm (now others underway)
- Slave nodes are doing *approximately similar* things for a given job
- Gather metrics and extract statistics
  - Determine metrics of relevance
  - For both black-box and white-box data
- Peer-compare histograms, means, etc. to determine "odd-man out"
- Extensions now to cover heterogeneity

Assumptions

- Majority of the system is working correctly
- Problems manifest as observable behavioral changes
  - Exceptions or performance degradations
  - Visible to the end-user
- All instrumentation is locally time-stamped
- Clocks are synchronized to enable system-wide correlation of data
- Instrumentation faithfully captures system behavior

Overview of Approach

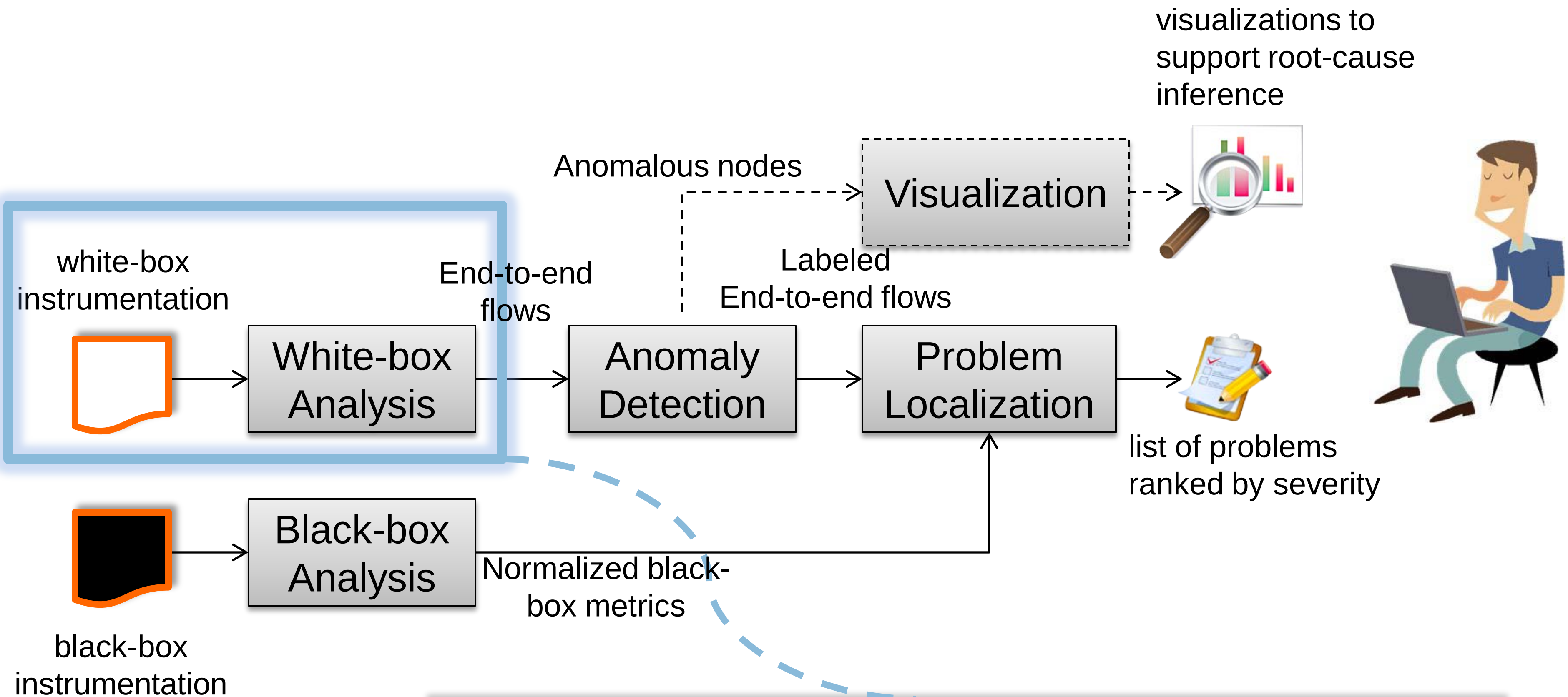


How About Those Metrics?

- White-box metrics (from Hadoop logs)
  - Event-driven (based on Hadoop's activities)
  - *Durations*
    - Map-task durations, Reduce-task durations, ReduceCopy-durations, etc.
  - System-wide dependencies between tasks and data blocks
  - Heartbeat information: Heartbeat rates, Heartbeat-timestamp skew between the Master and Slave nodes

- Black-box metrics (from OS /proc & Ganglia)
  - 64 different time-driven metrics (sampled every second)
  - Memory used, context-switch rate, User-CPU usage, System-CPU usage, I/O wait time, run-queue size, number of bytes transmitted, number of bytes received, pages in, pages out, page faults

White-Box Analysis



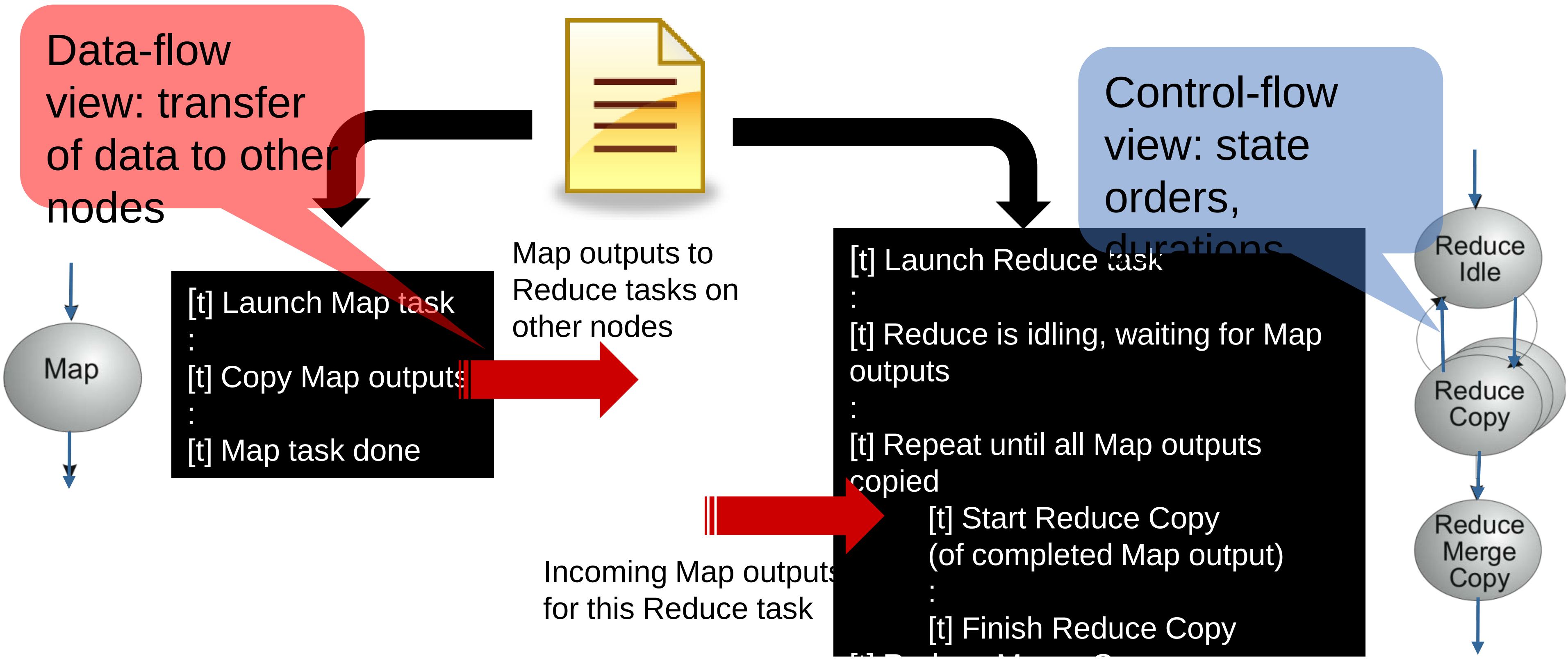
Questions

- How do we extract local control- and data-flow?
- How do we infer dependencies with other components?
- How do we deal with missing dependency information?

White-Box Analysis

- SALSA: Analyzing Logs as StAte Machines [*USENIX WASL 2008*]
- Extract state-machine views of execution from Hadoop logs
  - Distributed control-flow view of logs
  - Distributed data-flow view of logs
- Diagnose failures based on statistics of these extracted views
  - Control-flow based diagnosis
  - Control-flow + data-flow based diagnosis
- Perform analysis incrementally so that we can support it online a

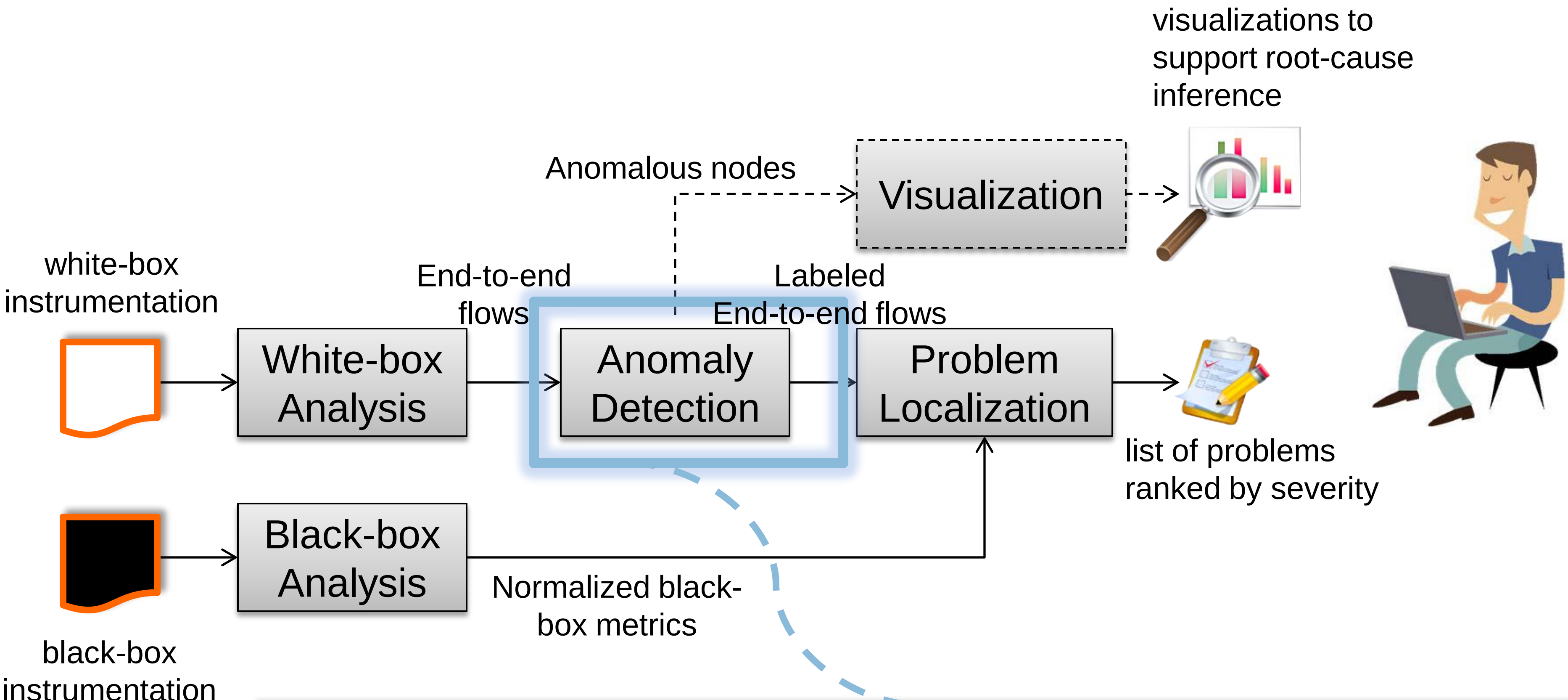
White-Box Analysis for Hadoop



Distributed Control+Data Flow

- Distributed control-flow
  - Causal flow of task execution across cluster nodes, i.e., Reduces waiting on Maps via Shuffles
- Distributed data-flow
  - Data paths of Map outputs shuffled to Reduces
  - HDFS data blocks read into and written out of jobs
- Job-centric causal flow: Fused Control+Data Flows
  - Correlate paths of data and execution
  - Create conjoined causal paths from data source before, to data destination after, processing

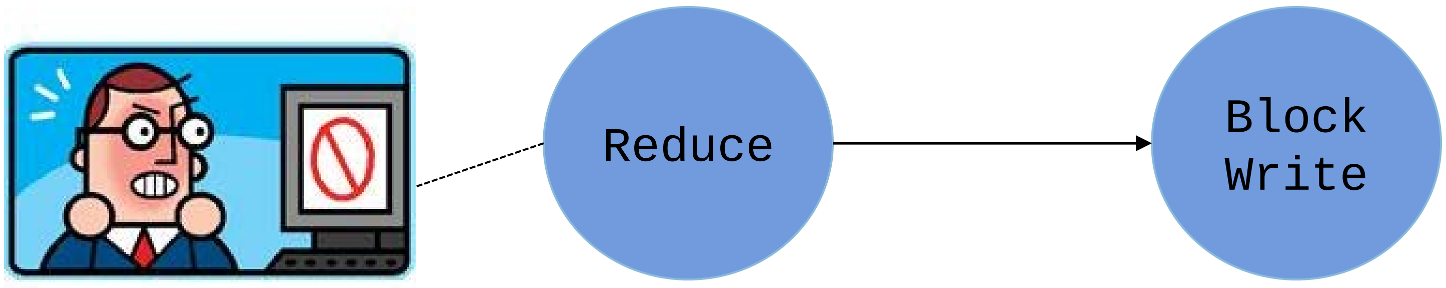
Anomaly Detection



Questions

- How to detect performance problems in the absence of labeled data?
- How to distinguish legitimate application behavior vs. problems?

Anomaly Detection



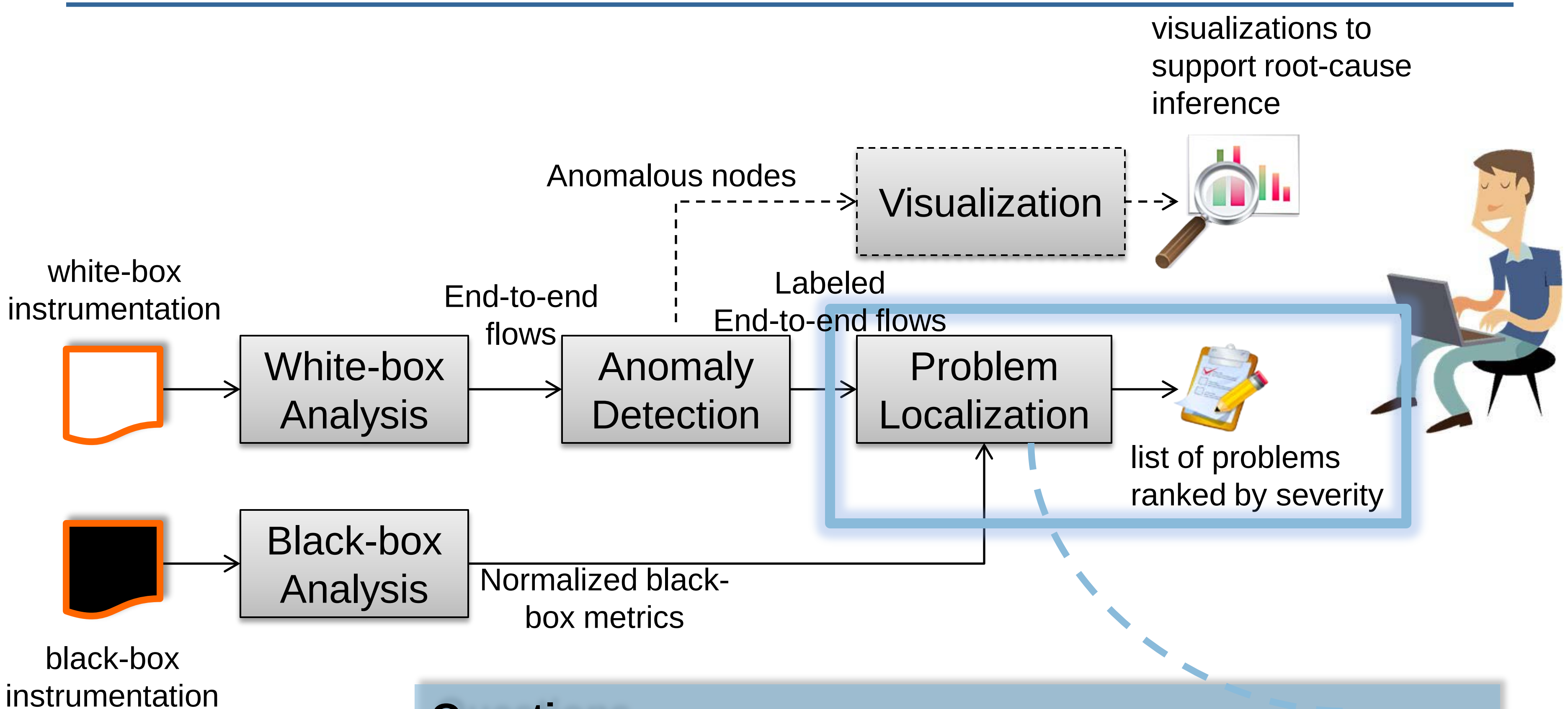
- Some user-visible problems manifest as errors
  - Detected by extracting error codes from failed flows, or
  - Apply domain-specific heuristics
- Performance problems can be harder to detect
  - Exploit the notions of “peers” to detect performance problems
  - Determine what system behaviors can be considered equivalent (“peers”) under normal conditions
  - Significant deviation from “peers” is regarded anomalous

rika (Swahili), *noun*. peer, contemporary, age-set, undergoing rites of passage (marriage) at similar times.

Anomaly Detection (1)

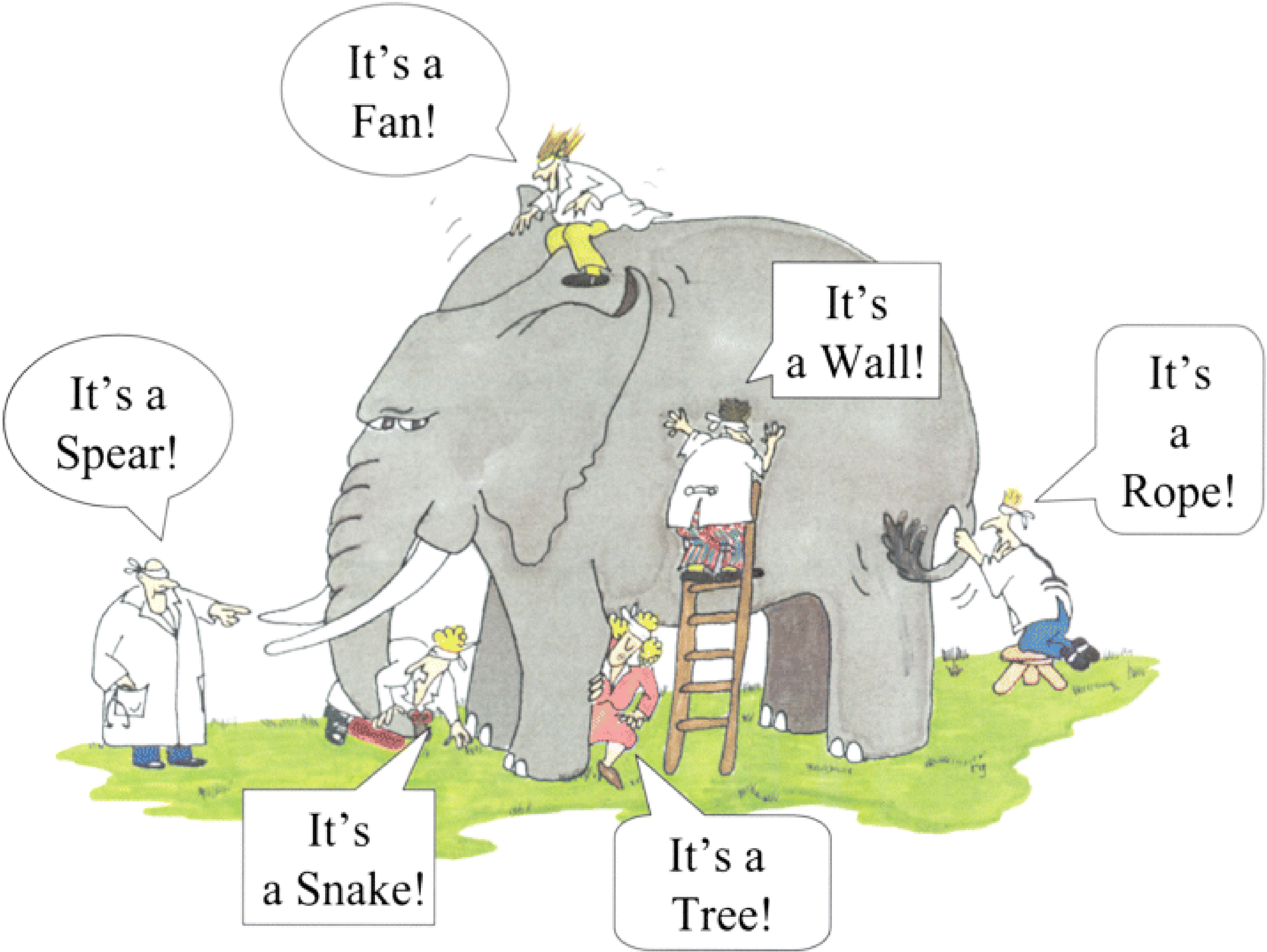
- Detect performance problems using “peers”
  - Empirical analysis of production data to identify peers
    - 219,961 successful jobs (Yahoo! M45 and OpenCloud)
    - 89% of jobs had low variance in their Map durations
    - 65% of jobs had low variance in their Reduce durations
  - Designate tasks belonging to the same job as peers
- At the same time, behavior amongst peers can legitimately diverge due to various application factors
  - Identified 12 such factors on OpenCloud
  - Example: HDFS bytes written/read

Problem Localization



Questions

- How to identify problems due to combination of factors?
- How to distinguish multiple ongoing problems?
- How to find resource that caused the problem?
- How to handle “noise” due to flawed anomaly detection?


It's a Fan!
It's a Spear!
It's a Wall!
It's a Rope!
It's a Snake!
It's a Tree!

Fusing the Metrics

JobTracker Durations views

TaskTracker Durations views

Job-centric data flows

TaskTracker heartbeat timestamps

JobTracker heartbeat timestamps

Black-box resource usage

Fusing Black-box Metrics

Determine if resource-usage metrics affected

Annotate flows associated with culprit nodes (and peers)

Culprit Node	Peer	Peer
Server 8	**Server 10**	**Server 13**
Time: 10:03:59, Map ID: task_188_m_98 Bytes Read: 7867 Duration: 25 seconds Status: FAILED	Time: 10:03:59, Map ID: task_188_m_76 Bytes Read: 7867 Duration: 3 seconds Status: SUCCESS	Time: 10:03:59, Map ID: task_188_m_85 Bytes Read: 6863 Duration: 2 seconds Status: SUCCESS
Mean CPU: 70.4% Mean Memory: 500MB Mean DiskUtil: 30KB	Mean CPU: 12.4% Mean Memory: 430MB Mean DiskUtil: 32KB	Mean CPU: 15.4% Mean Memory: 480MB Mean DiskUtil: 23KB

Mean resource-usage on node during event duration

Experimental Evaluation

		HADOOP
FAULT INJECTION	Workload	Gridmix cluster benchmark
	Injected faults	Resource hogs/Task hangs 10 iterations per fault
	Experimental setup	10-node EC2 cluster 2 1.2GHz cores, 7GB RAM
CASE STUDIES	Production Sytem	OpenCloud
	Status	Post-mortem offline analysis of real incidents

Impact of Fusion

QUESTION: Does fusion of metrics provide insight on root-cause?

METHOD: Hadoop EC2 cluster, 10 nodes, fault injection.

- Apply problem localization with fused white/black-box metrics.

	Top Metrics Indicted		Insight on root-cause
Fault Injected	**White box**	**Black-box**	
Disk hog	Maps	Disk	✓
Packet-loss	Shuffles	-	✗
Map hang (Hang1036)	Maps	-	✓
Reduce hang (Hang1152)	Reduces	-	✓

Fusion of metrics provides insight on most injected faults

Case: Multiple Hardware Issues

INCIDENT: Multiple hardware problems in OpenCloud cluster

- User experiences multiple job failures with cryptic exceptions.
- Administrators initially suspected memory configuration issue.
- Took a week to resolve. Bad disk and bad NIC on two nodes.

DIAGNOSIS APPLIED

- Apply problem-localization approach with white-box metrics.
- Correctly identified nodes with bad hardware in top-10 ranked list

Identified multiple simultaneous problems affecting user's job.

Lessons Learned (1)

- Synthesis of end-to-end causal traces possible
  - Local logs capture local control- and data-flow info
  - Inferring implicit dependencies
- In absence of labeled data, peer-comparison is feasible approach for anomaly detection
  - Peers can be tasks (Hadoop), end-to-end flows
- Regression can help to differentiate between
  - Legitimate application behavior (more bytes read/written) vs. anomalous behavior (task taking longer to run for other unexplained reasons)

Lessons Learned (2)

- Important to analyze both successful and failed flows
  - Limiting analysis to only failed flows might elevate common elements over causal elements
- Fusion of white+black-box data can provide more insight into source of problem
- Ranking problems by severity helps tolerate noise
  - Spurious labels receive lower ranking

Limitations

- No diagnosis for the Master node of a Hadoop cluster
  - Problems at master typically result in system-wide issues
- Peer-groups are defined statically
  - Need to automate identification of peers
- False positives occur if root-cause not in logs
  - Algorithm tends to implicate adjacent network elements
  - Need to incorporate more data to improve visibility
- Does not detect dormant problems that do not impact user-perceived system behavior
  - Examples: Blacklisted nodes in Hadoop

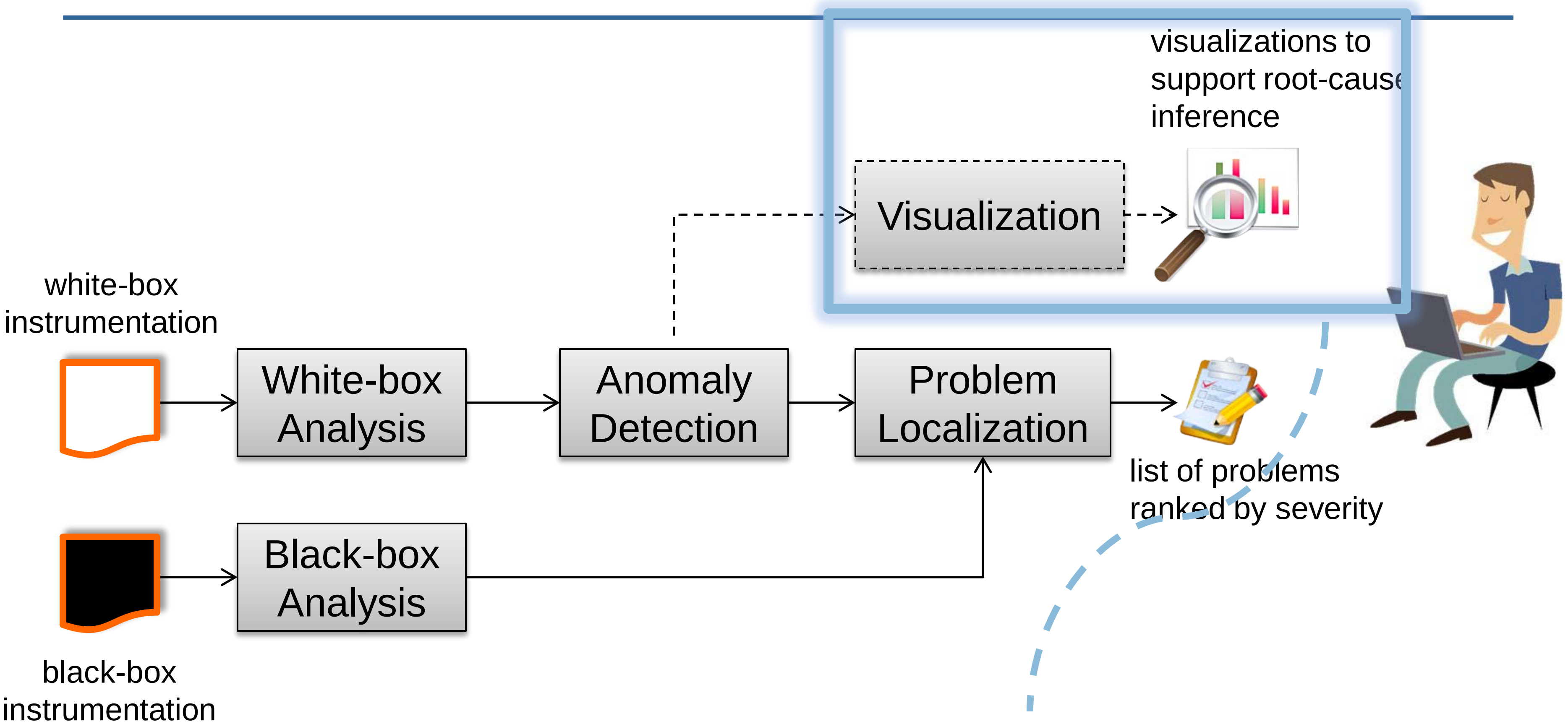
Extensions (Future Work)

- Visualization in heterogeneous systems
  - ✓ User study on diagnosis interfaces in Hadoop [CHIMIT11]
  - ✓ Visual signatures of problems in Hadoop [LISA12]
  - ✗ Visual signatures of problems in heterogeneous systems
  - ✗ Extensible visualization framework for diagnosis

Future Work

- Online monitoring and diagnosis
  - ✓ Generic framework for monitoring and diagnosis [WADS09]
  - ✓ Streaming implementation of problem-localization [DSN12]
  - ✗ Scalable monitoring and diagnostic framework

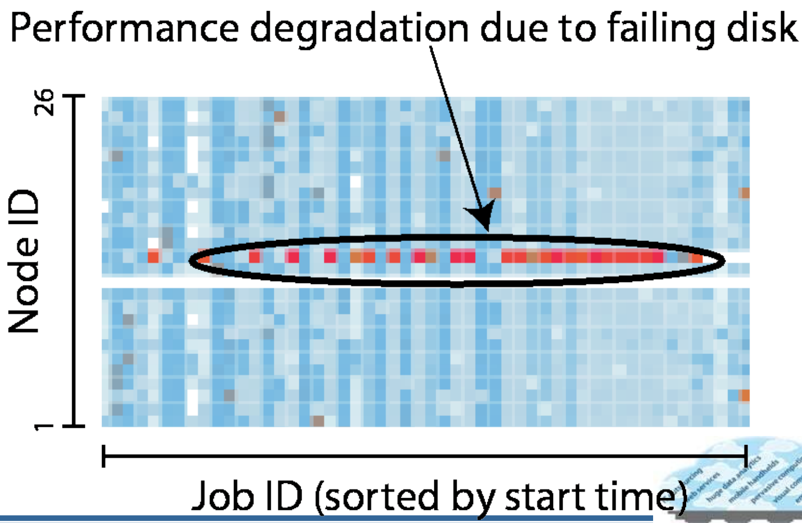
Visualization



Questions
- How to develop compact visualizations for large clusters?
- Can visualizations help spot/discriminate different anomalies?

Theia: Visual Signatures of Problems

- Maps anomalies observed to broad problem classes
  - Hardware failures, application issue, data skew
- Supports interactive data exploration
  - Users drill-down from cluster- to job-level displays
  - Hovering over the visualization gives more context
- Compact representation for scalability
  - Can support clusters with 100s of nodes



*USENIX LISA 2012 Best Student-Paper Award

Conclusion

- Approach for diagnosis of performance problems
  - Amenable for use in production systems
  - Infers dependencies from existing white-box logs
  - Uses heuristics and peer-comparison to detect anomalies
  - Localizes source of problem using statistical approach
  - Incorporates both white-box and black-box logs
- Demonstrated for two production systems
  - VoIP system at ISP (approach deployed for 2 years now)
  - OpenCloud Hadoop cluster
- Initial progress on extensions (visualization)

Publications (1)

Diagnosis in VoIP

1. S. P. Kavulya, S. Daniels, K. Joshi, M. Hiltunen, R. Gandhi, P. Narasimhan. Draco: Statistical Diagnosis of Chronic Problems in Large Distributed Systems. IEEE Dependable Systems and networks (DSN'12), Boston, MA, Jun 2012.
2. S. P. Kavulya, K. Joshi, M. Hiltunen, S. Daniels, R. Gandhi, P. Narasimhan. Practical Experiences with Chronics Discovery in Large Telecommunications Systems. Best Papers from SLAML in Operating Systems Review (OSR'12), 2012.
3. S. P. Kavulya, K. Joshi, M. Hiltunen, S. Daniels, R. Gandhi, P. Narasimhan. Practical Experiences with Chronics Discovery in Large Telecommunications Systems. Workshop on Managing Large-Scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques (SLAML'11), 2011.

Visualization, User studies, Surveys

4. E. Garduno, S. Kavulya, J. Tan, R. Gandhi and P. Narasimhan. Theia: Visual Signatures for Problem Diagnosis in Large Hadoop Clusters. In Large Installation System Administration Conference (LISA) 2012, San Diego, CA, Dec 2012. *Best Student Paper Award.*
5. S. P. Kavulya, K. Joshi, F. Di Giandomenico, P. Narasimhan. Failure Diagnosis of Complex Systems.Book on Resilience Assessment and Evaluation (RAE'12). Wolter, 2012.
6. J. Campbell, A. Ganesan, B. Gotow, S. Kavulya, J. Mulholland, P. Narasimhan, S. Ramasubramanian, M. Shuster, and J. Tan. Understanding and Improving the Diagnostic Workflow of MapReduce Users. In 5th ACM Symposium on Computer Human Interaction for Management of Information Technology (CHIMIT), Boston, MA, Dec 2011.
7. S. Kavulya, J. Tan, R. Gandhi, P. Narasimhan. An Analysis of Traces from a Production MapReduce Cluster. 10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid) 2010, Melbourne, Victoria, Australia, May 2010.

White-box diagnosis

8. J. Tan, S. Kavulya, R. Gandhi, P. Narasimhan. Visual, Log-based Causal Tracing for Performance Debugging of MapReduce Systems. 30th IEEE International Conference on Distributed Computing Systems (ICDCS) 2010, Genoa, Italy, Jun 2010.
9. J. Tan, X. Pan, S. Kavulya, R. Gandhi, P. Narasimhan. Mochi: Visual Log-Analysis Based Tools for Debugging Hadoop. USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '09), San Diego, CA, Jun 2009.
10. J. Tan, X. Pan, S. Kavulya, R. Gandhi, P. Narasimhan. SALSA: Analyzing Logs as State Machines. USENIX Workshop on Analysis of System Logs (WASL'08), San Diego, CA, Dec 2008.

Black-box diagnosis

11. J. Tan, S. Kavulya, R. Gandhi, P. Narasimhan. Lightweight Black-box Failure Detection for Distributed Systems. In Workshop on Management of Big Data systems (MBDS) 2012, co-located with the International Conference on Autonomic Computing, San Jose, SA, Sep 2012.
12. X. Pan, S. Kavulya, J. Tan, R. Gandhi, P. Narasimhan. Ganesha: Black-Box Diagnosis for MapReduce Systems. Workshop on Hot Topics in Measurement & Modeling of Computer Systems (HotMetrics), Seattle, WA, Jun 2009.

Publications (2)

Black-box + White box diagnosis

13. J. Tan, X. Pan, S. Kavulya, E. Marinelli, R. Gandhi, P. Narasimhan. Kahuna: Problem Diagnosis for MapReduce-based Cloud Computing Environments. 12th IEEE/IFIP Network Operations and Management Symposium (NOMS) 2010, Osaka, Japan, Apr 2010.
14. X. Pan, J. Tan, S. Kavulya, R. Gandhi, P. Narasimhan. Blind Men and the Elephant: Piecing Together Hadoop for Diagnosis. 20th IEEE International Symposium on Software Reliability Engineering (ISSRE), Industrial Track, Mysuru, India, Nov 2009.
15. S. Kavulya, R. Gandhi, P. Narasimhan. Gumshoe: Diagnosing Performance Problems in Replicated File-Systems. IEEE Symposium on Reliable Distributed systems (SRDS'08), Naples, Italy, October 2008.
16. S. Pertet, R. Gandhi, P. Narasimhan. Fingerpointing Correlated Failures in Replicated Systems. SysML, April 2007.

Students

- Soila Kavulya – now at Intel Labs
- Jiaqi Tan
- Nathan Mickulicz
- Utsav Drolia
- Mike Kasick – graduating early 2014
- Rolando Martins – post-doctoral researcher