Theia: Visual Signatures for Problem Diagnosis in Large Hadoop Clusters

Elmer Garduno, Soila P. Kavulya, Jiaqi Tan, Rajeev Gandhi, Priya Narasimhan
Carnegie Mellon University

Abstract

Diagnosing performance problems in large distributed systems can be daunting as the copious volume of monitoring information available can obscure the root-cause of the problem. Automated diagnosis tools help narrow down the possible root-causes—however, these tools are not perfect thereby motivating the need for visualization tools that allow users to explore their data and gain insight on the root-cause. In this paper we describe Theia, a visualization tool that analyzes application-level logs in a Hadoop cluster, and generates visual signatures of each job's performance. These visual signatures provide compact representations of task durations, task status, and data consumption by jobs. We demonstrate the utility of Theia on real incidents experienced by users on a production Hadoop cluster.

1 Introduction

Hadoop [29] is an open-source implementation of Google's MapReduce [9] framework. As of 2012, a typical Hadoop deployment consisted of tens to thousands of nodes [30, 31]. Manual diagnosis of performance problems in a Hadoop cluster requires users to comb through the logs on each node—a daunting task, even on clusters consisting of tens of nodes. Fortunately, there has been significant research on automated diagnosis in distributed systems; ranging from techniques that generate and analyze end-to-end causal traces [14, 25], to alarm-correlation techniques [13, 20, 16].

In recent years, there has also been an increased interest in developing diagnosis techniques that understand the application-specific semantics of MapReduce jobs, thereby providing users with insight into the behavior of their jobs [8, 28, 2]. However, the use of automated diagnosis techniques in isolation is not always sufficient to localize problems at the level of granularity desired by users. Visualization tools help bridge this gap by providing interactive interfaces that allow users to explore their data, and formulate their own hypothesis about the root-cause of problems. A number of visualization tools focus on visualizing time-series data [17, 18], request flows [26, 10], and the outputs of automated diagnosis algorithms [15].

Our tool, Theia[1], leverages application-specific semantics about the structure of the MapReduce programming model to generate high-density, interactive visualizations of job performance that scale to support current industry deployments. A recent study of users at a production Hadoop cluster [5] highlighted users' need to differentiate application-level problems (e.g., software bugs, workload imbalances) from infrastructural problems (e.g., contention problems, hardware problems). In Theia, we have developed *visual signatures* that allow users to easily spot performance problems due to application-level and infrastructural issues.

We developed three different visualizations: one at the cluster-level that represents the performance of jobs across nodes over time, and two others at the job-level that summarize task performance across nodes in terms of task duration, task status and volume of data processed. We evaluated these visualizations using real problems experienced by Hadoop users at a production cluster over a one-month period. Our visualizations correctly identified 192 out of 204 problems that we observed during that time period.

The rest of the paper is organized as follows. Section 2 describes the Hadoop production cluster, and the challenges of diagnosing problems at scale. Section 3 describes the design and implementation of our visualization tool, Theia. Section 4 describes our visualizations using case studies drawn from real problems experienced by Hadoop users in the cluster. Finally, Section 5 and 6 present related work and conclusions.

[1] Named after the Greek goddess of light.

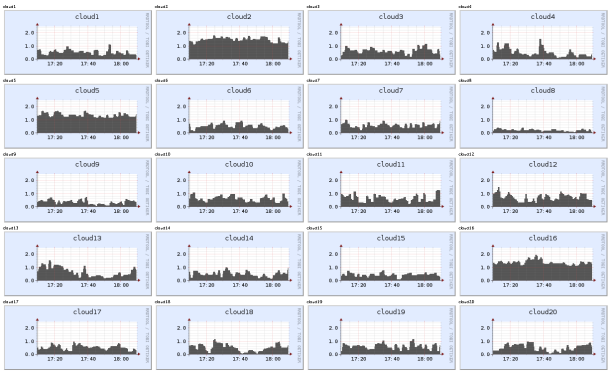

Figure 1: Screenshot from *Ganglia* where each chart represents load on a node during a one hour period.

2 Hadoop cluster, Users, and Challenges

Hadoop production cluster. We analyzed the jobs and problems experienced by Hadoop users at the OpenCloud cluster for data-intensive research. The OpenCloud cluster currently has 64 worker nodes, each with 8 cores, 16 GB DRAM, 4 1TB disks and 10GbE connectivity between nodes [22].

Each Hadoop job consists of a group of Map and Reduce tasks performing some data-intensive computation. Hadoop automates job scheduling and allows multiple users to share the cluster. The master node in a Hadoop cluster typically runs two daemons: 1) the JobTracker that schedules and manages all of the tasks belonging to a running job; and 2) the NameNode that manages the Hadoop Distributed Filesystem (HDFS) namespace by providing a filename-to-block mapping, and regulating access to files by clients. Each slave node runs two daemons: 1) the TaskTracker that launches and tracks the progress of tasks on its local node; and 2) the DataNode that serves data blocks to HDFS clients.

We analyzed one-month's worth of logs generated by Hadoop's JobTracker on the OpenCloud cluster. These logs store information about the Map and Reduce tasks executed by each job. This information comprises of start times, durations, status, volume of data read and written, and user-generated performance counters for each task.

Users. The users of the cluster are researchers familiar with configuring Hadoop, writing and running MapReduce jobs, and analyzing the output generated by their jobs. These users run a diverse set of data-intensive workloads such as large-scale graph mining, text and web mining, natural language processing, machine translation problems, and data-intensive file system applications. The Hadoop users try to diagnose problems either on their own, by soliciting help from the cluster mailing list, or by escalating the problem to the system administrators. The users are primarily interested in distinguishing between two diagnostic scenarios [5]: 1) application-level problems (e.g., software bugs, workload imbalances) which they can fix on their own, and 2) infrastructural problems (e.g., contention problems, hardware problems) which they should escalate to the system administrators.

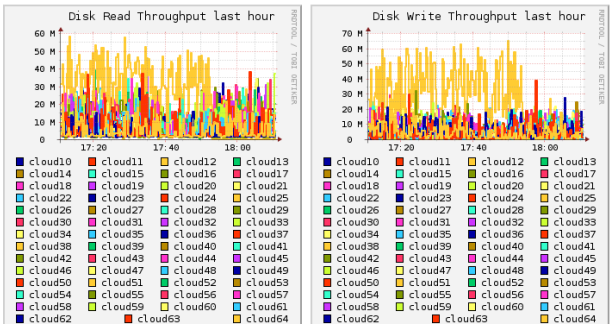

Figure 2: Screenshot of disk throughput where each line represents a node's throughput during a one hour period.

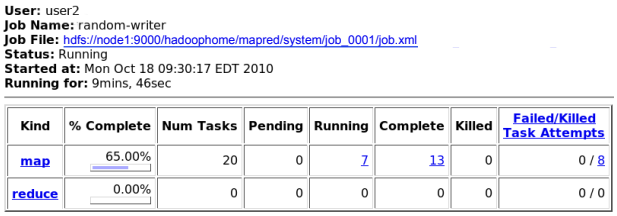

Figure 3: Screenshot of the Hadoop web interface showing progress of Map and Reduce tasks in a single job.

Visualization challenges. Analyzing logs from Hadoop clusters can be a daunting task due to the large amounts of information available [23, 27]. For example, in the OpenCloud cluster, we have observed jobs with over 120,000 tasks running on over 60 nodes. This information overload would be compounded in larger clusters supporting thousands of nodes. Widely-used tools such as Ganglia [17] and Nagios [19] which allow users to monitor the performance of nodes on their clusters, do not fully capture the semantics of MapReduce applications—which are important when trying to diagnose application-level problems.

In addition, the time-series representation of node-level performance generated by these tools is not scalable. The problem of visualizing information from a large number of nodes is exemplified in Figure 1 which displays Ganglia's visualization of load across multiple

Table 1: Heuristics for developing visual signatures of problems experienced in a Hadoop cluster.

Dimension	Visual Signatures		
	Application problem	**Workload imbalance**	**Infrastructural problem**
Time	*Single* user or job over time	*Single* user or job over time	*Multiple* users and jobs over time
Space	Span *multiple* nodes	Span *multiple* nodes	Typically affect *single* node, but correlated failures also occur
Value	Performance degradations and *task exceptions*	Performance degradation and *data skews*	Performance degradations and *task exceptions*

nodes on single screen. If the number of nodes is large, users will have a difficult time correlating the load across nodes. An attempt to ease correlation across nodes by displaying every node's time-series data in a single chart also leads to information overload, as shown in Figure 2 which displays disk throughput across multiple nodes.

Hadoop provides a web interface that allows users to browse application logs and monitor the progress of the Map and Reduce tasks that constitute their jobs. Hadoop also provides a separate interface where users can monitor the status of the distributed filesystem. However, these interface can become bloated for clusters with large numbers of jobs and tasks since users have to scroll through long lists of tasks, and click through multiple screens to troubleshoot a problem. The information displayed by the web interface, as shown in Figure 3, is a snapshot of the current state of a job—making it difficult to discern historical trends in performance.

3 Theia: Visual Signatures for MapReduce

We developed Theia—a visualization tool that characterizes the (mis-)behavior of large MapReduce clusters as a series of visual signatures, and facilitates troubleshooting of performance problems on the cluster. We targeted performance problems due to hardware failures or data skews, and failed jobs due to software bugs. Our current implementation does not address performance problems due to misconfigurations. The key requirements for Theia were: 1) an interactive interface that supports data exploration thereby enabling users to formulate a hypothesis on the root-cause of problems; and 2) compact representations that can support MapReduce clusters consisting of up to thousands of nodes.

We implemented Theia using a Perl script that gathered data about job execution from the job-history logs generated by Hadoop's JobTracker. We stored this information in a relational database, and generated visualizations in the web browser using the D3 framework [4].

We developed visual signatures that allow users to spot of patterns (or signatures) of misbehavior in job execution by identifying visual patterns across the time, space, and value domain. Table 1 summarizes the heuristics that we used to develop visual signatures that distinguish between application-level problems, workload imbalances between tasks from the same job, and infrastructural problems.

We developed the visualizations iteratively by manually identifying jobs which we knew had failed, and consulting with the system administrators to learn what incidents had occurred in the cluster. Next, we developed the visualizations using a subset of these incidents, and iterated through different designs to select the visualizations that best displayed the problem. We then manually verified that the visualizations matched up with the heuristics for distinguishing between different problems. These heuristics are explained below:

1. **Time dimension.** Different problems manifest in different ways over time. For example, application-level problems and workload imbalances are specific to an application; therefore, the manifestation of a problem is restricted to a single user or job over time. On the other hand, infrastructural problems, such as hardware failures, affect multiple users and jobs running on the affected nodes over time.

2. **Space dimension.** The space dimension captures the manifestation of the problem across multiple nodes. Application-level problems and workload imbalances associated with a single job manifest across multiple nodes running the buggy or misconfigured code. Infrastructural problems are typically limited to a single node in the cluster. However, a study of a globally distributed storage system [11] shows that correlated failures are not rare, and were responsible for approximately 37% of failures. Therefore, infrastructural problems can also span multiple nodes.

3. **Value dimension.** We quantify anomalies in the value domain by capturing the extent of performance degradation, data skew, and task exceptions experienced by a single job. Application-level and infrastructural problems manifest as either performance degradations or task exceptions. Workload imbalances in Hadoop clusters can stem from skewed data distributions that lead to performance degradations.

3.1 Quantifying Anomalies

To generate the visual signatures of problems, we quantify the anomalies experienced during job execution using a small number of metrics. We visualize *Map* and *Reduce* tasks separately because these tasks have very different semantics. We detect anomalies by first assuming that under fault-free conditions, the workload in a Hadoop cluster is relatively well-balanced across nodes executing the same job—therefore, these nodes are peers and should exhibit similar behavior [21]. Next, we identify nodes whose task executions differ markedly from their peers and flag them as anomalous. Aggregating task behavior on a per-node basis allows us to build compact signatures of job behavior because the number of nodes in the cluster can be several orders of magnitudes smaller than the maximum number of tasks in a job. We flag anomalous nodes based on the following metrics:

1. **Task duration.** Task duration refers to the span of a task execution for a given job on a single node. Performance degradations are detected by identifying nodes whose task durations significantly exceed those of its peers.

2. **Data volumes.** The volume of data processed is the total number of bytes read or written from the local filesystem, and the Hadoop Distributed Filesystem (HDFS). We flag anomalies when nodes process significantly more data than their peers (indicating a workload imbalance) or significantly less data (possibly indicating performance problem at the node).

3. **Failure ratios** The failure ratio is computed by aggregating tasks on a per-node basis, and calculating the ratio of failed to successful tasks. In our visualizations, we distinguish *failed* tasks from *killed* tasks which arise when speculatively-executed tasks are terminated by the task scheduler.

To compute the anomaly score we assume that metrics follow a normal distribution and use the z-score, a dimensionless quantity that indicates how much each value deviates from the mean in term of standard deviations, and is computed using the following formula: $z = \frac{x-\mu}{\sigma}$, where μ is the mean of the values, and σ is the corresponding standard deviation.

For the cluster-level visualization, we estimate the severity of problems by using a single anomaly score that flags nodes as anomalous if the geometric mean of the absolute value of the z-scores is high, *i.e.*, $AnomalyScore = (|z_{TaskDuration}| * |z_{DataVolume}| * |z_{FailureRatio}|)^{(1/3)}$.

Table 2: Metrics utilized by the visualizations.

Metric	Type
Duration	Scalar/Anomaly
Information volume	Scalar/Anomaly
Successful tasks	Count
Killed tasks	Count
Failed tasks	Count
Data skew	Percentage
Job name & job id	Text
Date & time	Text

3.2 Visualizing Anomalies

To generate visualizations that are meaningful in clusters with hundreds or thousands of nodes, we take advantage of the human brain's deduction and perception capabilities [12]. Human perception is determined by two kinds of processes: bottom-up, driven by the visual information in the pattern of light falling on the retina, and top-down, driven by the demands of attention, which in turn are determined by the needs of the tasks. [33]. In our case, the top-down task is to find those nodes, tasks, or jobs that are exhibiting anomalous behavior. We have generated three visualizations that represent different aspects of a job's execution at varying levels of granularity.

All of the visualizations in our system are designed with the following guidelines: 1) they display jobs and nodes in an order that preserves contextual information, e.g., sorting nodes by the amount of data they process; 2) they clearly distinguish between attributes that have different semantics, *e.g.*, distinguishing between Map or Reduce tasks, and failed and killed tasks; and 3) they preserve the structure of the information across the different visualizations. We use a square as the unit of representation for the different metrics of the tasks executing on a node. Table 2 shows the different metrics represented in our visualizations.

We also made the following design decisions: 1) present as much information as is understandable in a single viewport by using color and size to signal the relevant information—this allows the brain's visual query mechanism to process large chunks of information; 2) postpone the display of non-relevant attributes and take advantage of the interactive nature of web-browsers—all of our visualizations provide access to additional information by using the mouseover gesture, and allowing users to drill-down to a more detailed view of the data by clicking on the relevant interface elements.

3.3 Scalability

A Hadoop cluster might be composed of hundreds or thousands of nodes—leading to challenges in problem

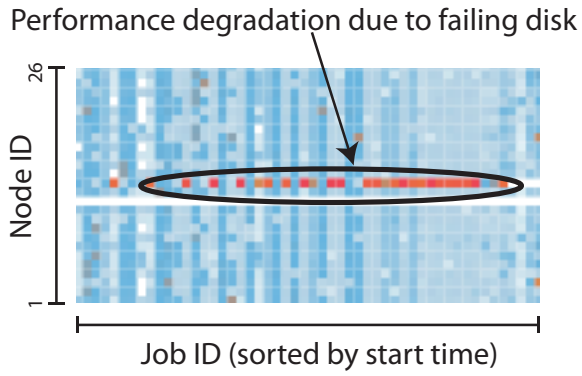

Figure 4: Visual signature of an *infrastructural problem* shows succession of anomalous jobs (darker color) due to a failing disk controller on a node.

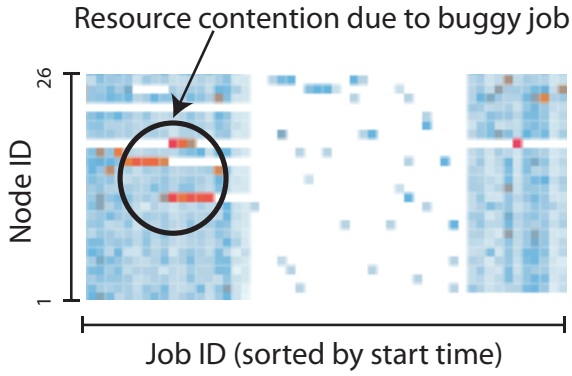

Figure 5: Visual signature of an *application-level problem*. In this case, the problem was caused by a single user submitting a resource-intensive job to the cluster repeatedly causing degraded performance on multiple nodes which were eventually blacklisted.

diagnosis brought about by the scale of the system. Research on human perception has shown that the brain can manage to distinguish features at a very high resolution, *e.g.*, differentiating up to 250 features per linear inch [32]. The following formula can be used to calculate the data density of a visualization:

$$\text{Data density} = \frac{\text{Number of data entries or features}}{\text{Area of data display}}$$

For scalability, we leveraged *high-resolution data graphics* to display the relevant information of each node in the cluster. Our heatmap visualization is capable of representing between 1,500 and 2,900 features per square inch, depending on the resolution of the display; this contrasts with the approximately 10 features per square inch shown in a typical publication [32].

4 Visualizations and Case Studies

We developed three different visualizations that facilitate problem diagnosis. We describe their design considerations, and use cases in this section. The first visualization is the anomaly heatmap which summarizes job behavior at the cluster-level; the other two visualizations are at the job-level. The first job-level visualization, referred to as the job-execution stream, allows users to scroll through jobs sequentially thus preserving the time context. The second job-level visualization, referred to as the job-execution detail, provides a more detailed view of task execution over time on each node in terms of task duration and amount of data processed.

4.1 Anomaly Heatmap

A heatmap is a high-density representation of a matrix, that we use to provide users with a high-level overview of jobs execution at the cluster-level. This visualization is formulated over a grid that shows nodes on the rows and jobs on the columns, as shown in Figure 4. The darkness of an intersection on the grid indicates a higher degree of anomaly on that node for that job. By using this visualization, anomalies due to application-level and infrastructural problems can be easily spotted as bursts of color that contrast with non-faulty nodes and jobs in the background.

Figure 4 displays the visual signature of an *infrastructural problem* identified by a succession of anomalous jobs (darker color) due to a failing disk controller on a node. Figure 5 illustrates the visual signature of an *application-level problem* caused by a single user repeatedly submitting a resource-intensive job to the cluster. The resource contention led to degraded performance across multiple nodes which were eventually blacklisted. It is easy to identify when a node has gone offline or has been blacklisted by spotting a sudden sequence of horizontal white-space; vertical white-space indicates periods of time when the cluster was idle.

The data density of the anomaly heatmap is around 2,900 features per square inch on a 109 ppi [2] display, using 2x2 pixels per job/node. This gives us a capacity to show approximately $54 \times 54 \approx 2900$ features per square inch [3]; which is equivalent to fit 1200 jobs x 700 nodes on a $27''$ display.

[2] ppi: pixels per inch
[3] (109*109ppi)/(2*2pixels) = 2970.25 features per square inch

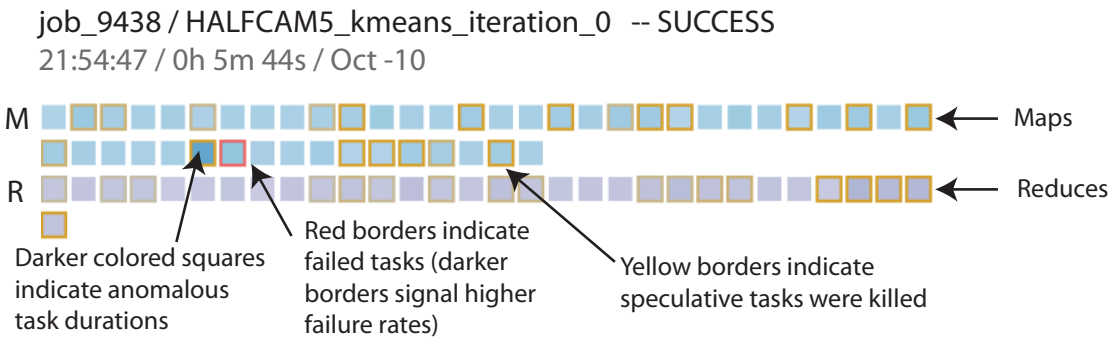

Figure 6: The *job execution stream* visualization compactly displays information about a job's execution. The header lists the job ID, name, status, time, duration and date. The visualization also highlights anomalies in task duration by using darker colors, and task status by using yellow borders for killed tasks and red borders for failed tasks. The nodes are sorted by decreasing amount of I/O processed.

Figure 7: Visual signature of *bugs in the Map phase*. Failures spread across all Map nodes (solid red border) typically signals a bug in the Map phase.

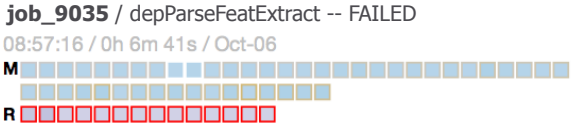

Figure 8: Visual signature of *bugs in the Reduce phase*. Failures spread across all Reduce nodes (solid red border) typically signals a bug in the Reduce phase.

Figure 9: Visual signature of *data skew*. A node with anomalous task durations (darker color) and high volume of I/O (first nodes in the list) can signal data skew.

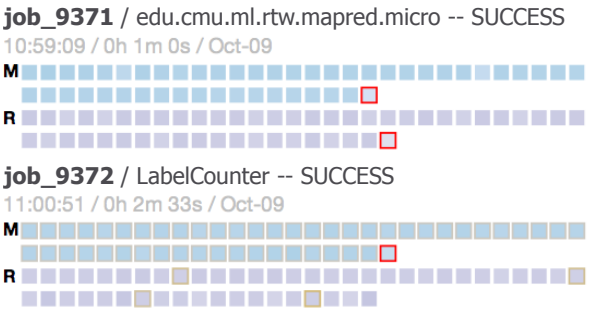

Figure 10: Visual signature of *infrastructural problem*. A node with failures (red border) spread across multiple jobs can signal problems with that node.

4.2 Job Execution Stream

The job execution stream, shown in Figure 6, provides a more detailed view of jobs while preserving information about the context by showing a scrollable stream of jobs sorted by start time. In addition to displaying general information about the job (job ID, job name, start date and time, job duration) in the header, this visualization has a more complex grammar that allows the representation of an extended set of signatures.

Since the application-semantics of Map and Reduce tasks are very different, we divided nodes into two sets: the Map set and the Reduce set. Each of these sets represent nodes that executed Map or Reduce tasks for a given job sorted by decreasing I/O. The intuition behind the sort order is that nodes that process significantly less data than their peers might be experiencing performance problems. We enhanced the representation of each node with a colored border that varies in intensity depending the ratio of failed tasks to successful tasks, or the ratio of killed tasks to successful tasks; killed tasks arise when the task scheduler terminates speculative tasks that are still running after the fastest copy of the task completed. Killed tasks are represented using a yellow border, which is overloaded by a red border if there are any failed tasks.

The job execution stream visualization allows us to

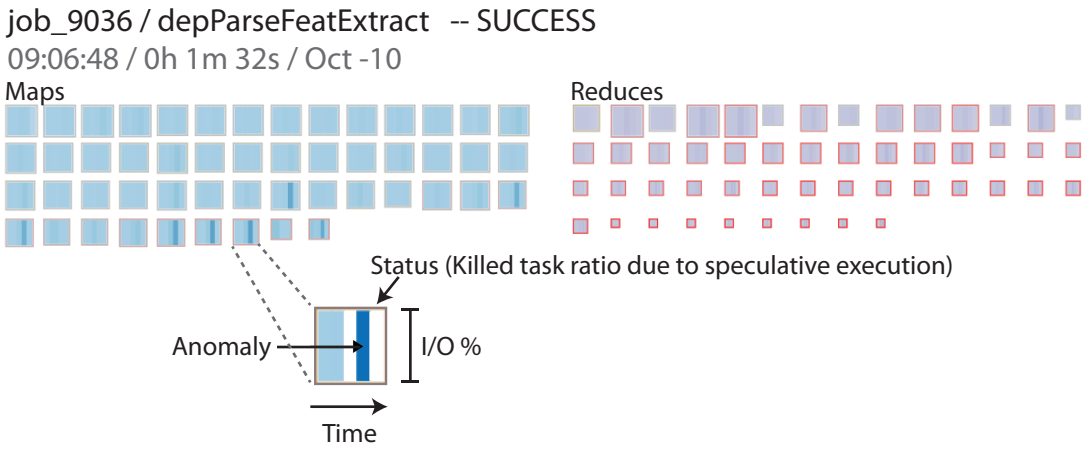

Figure 11: The *job execution detail* visualization highlights both the progress of tasks over time, and the volume of data processed. Each node is divided on five stripes that represent the degree of anomalies in tasks executing during the corresponding time slot; the size of the square represents the proportion of I/O processed by that node.

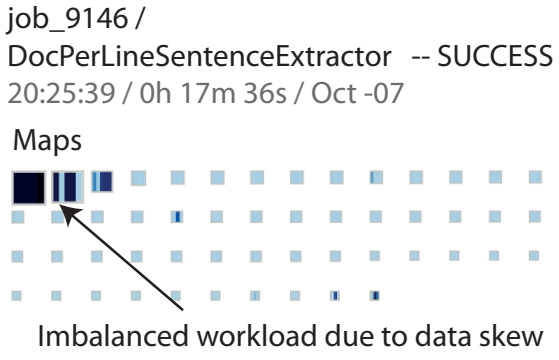

Figure 12: Visual signature of *data skew*. A single node with high duration anomaly (darker color) and high amount of I/O (first nodes in the list, large square size) can signal data skew.

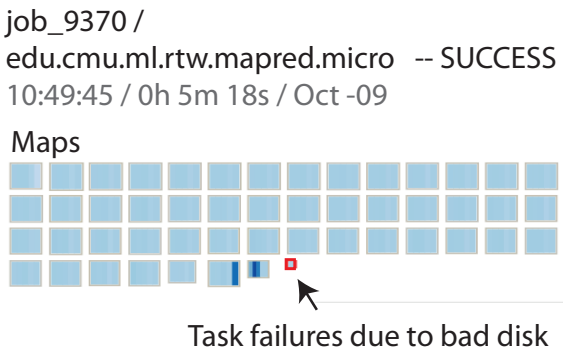

Figure 13: Visual signature of an *infrastructural problem*. A single node with high task durations (darker color) or failed tasks (red border), and a low volume of I/O (small square size) can indicate a hardware failure.

generate signatures for the following scenarios:

1. *Application-level problems.* Bugs in either the Map or Reduce phase manifest as a large number of failed tasks across all nodes in either the Map or Reduce set (see Figures 7 and 8).

2. *Workload imbalance or data skew.* A *data skew* can be identified by spotting nodes with anomalous task durations on the first positions of the node list. For example, see the dark left-most node in Figure 9.

3. *Infrastructural problems. Infrastructural problems* can be detected if a node recurrently fails across multiple jobs, as shown in Figure 10.

4.3 Job Execution Detail

The *job execution detail* visualization provides a more detailed view of task execution and is less compact than the *job execution stream*. The *job execution detail* visually highlights both the progress of tasks over time, and the volume of data processed as shown in Figure 11. Nodes are still represented as two sets of squares for Map and Reduce tasks; however, given that there is additional space available since we are only visualizing one job at a time, we use the available area on each of the squares to represent two additional variables: 1) anomalies in task durations over time by dividing the area of each node into five vertical stripes, each corresponding to a fifth of

Table 3: Problems diagnosed by cluster-level and job-level visualizations in Theia. The infrastructural problems consisted of 42 disk controller failures, 2 full hard drives, and 1 network interface controller (NIC) failure. The infrastructural problems diagnosed by the job execution stream were a subset of those identified by the heatmap.

Type	Total problems	Diagnosed by heatmap	Diagnosed by job execution stream
Application-level problem	157	0	157
Data-skew	2	2	2
Infrastructural problem	45	33	10

the total time spent executing tasks on that node—darker colors indicate the severity of the anomaly while white stripes represent slots of time where no information was processed; 2) percentage of total I/O processed by that node, *i.e.*, reads and writes to both the local filesystem and the Hadoop distributed filesystem (HDFS)—larger squares indicate higher volumes of data.

Figure 12 shows the visual signature of a data skew where a subset of nodes with anomalous task durations (darker color) and high amounts of I/O (first nodes in the list, large square size) indicate data skew. In this visualization, the data skew is more obvious to the user when compared to the same problem visualized using the job execution stream in Figure 9. Infrastructural problems as shown in Figure 13 can be identified as a single node with high task durations (darker color) or failed tasks (red border), coupled with a low volume of I/O (small square size) which might indicate a performance degradation. The data density of this visualization, using 24x24 squares and 7 features per square (5 time slots + I/O percentage + status ratio) on a 109 ppi screen, is about 112 numbers per square inch—this allows us to display up to 4800 nodes on a 27″ display (2400 nodes for Maps + 2400 nodes for Reduces).

4.4 Interactive User Interface

Theia is implemented as an interactive web interface supporting a top-down data exploration strategy that allows users to form, and confirm their hypotheses on the root-cause of the problems. Users can navigate from the cluster-level visualization to the job-level visualizations by clicking on the relevant interface elements. Theia takes 1–2 seconds to change between views. All of our visualizations provide access to additional information by using the mouseover gesture. Figure 14 uses the *job execution detail* visualization to show how the performance of a job was degraded by a failed NIC (Network Interface Controller) at a node highlighted using a dark blue square. By hovering over the failed node, a user can obtain additional information about the behavior of tasks executed on that node. For example, a user can observe that 50% of the tasks executed on this node failed.

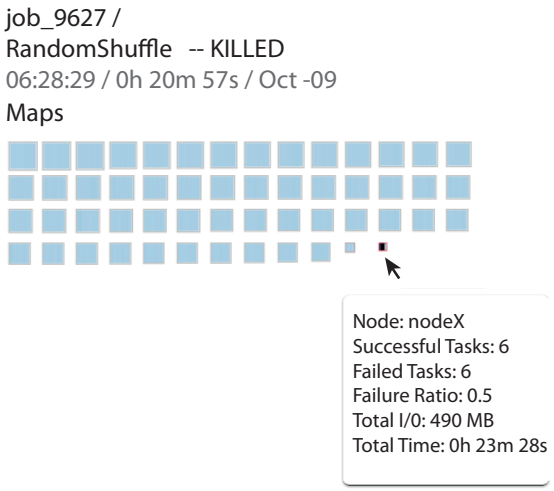

Figure 14: *Interactive User Interface*. This job execution detail visualization shows degraded job performance due to a NIC failure at a node. Hovering over the node provides the user with additional information about the behavior of tasks executed on that node.

4.5 Summary of Results

We generated our visualizations using one month's worth of logs generated by Hadoop's JobTracker on the OpenCloud cluster. During this period, 1,373 jobs where submitted, comprising of a total of approximately 1.85 million tasks. From these 1,373 jobs, we manually identified 157 failures due to application-level problems, and 2 incidents of data skew. We also identified infrastructural problems by analyzing a report of events generated by the Nagios tool installed on the cluster. During the evaluation period, Nagios reported 68 messages, that were associated with 45 different incidents namely: 42 disk controller failures, 2 full hard drives, and 1 network interface controller (NIC) failure.

We evaluated the performance of Theia by manually verifying that the visualizations generated matched up with the heuristics for distinguishing between different problems described in Table 1. Table 3 shows that we successfully identified all the application-level problems and data skews using the job execution stream (similar

results are obtained using the job execution detail). In addition, the anomaly heatmap was able to identify 33 of the 45 infrastructural problems (the problems identified by the job execution stream are a subset of those identified by the heatmap). We were unable to detect 4 of the infrastructural problems because the nodes had been blacklisted. We hypothesize that the heatmap was unable to detect the remaining 8 infrastructural problems because they occurred when the cluster was idle.

5 Related Work

The visualizations we generated are based on previous research on visual log-based performance debugging [27]. This paper presents a Host-State decomposition visualization from which we borrowed elements to build our Job-Node heatmap visualization. Bodik et al. [3] present a combined approach to failure localization using anomaly detection and visualization that leverages heatmaps. Theia also uses heatmaps to visualize anomalies, but we also present heuristics that allow users to distinguish between different types of problems.

HiTune [8], X-trace [10], and Dapper [26] are techniques for tracing causal request paths in distributed systems, but they also offer support for visualizing requests whose causal structure or duration are anomalous. While Theia does not currently support the visualization of the causal structure of Hadoop jobs, our visualizations provide a compact representation of multiple aspects of job behavior such as task duration, task status, and data volume within a single viewport.

Artemis [7] provides a pluggable framework for distributed log collection, data analysis, and visualization. LiveRAC [18] is tool for browsing and correlating time-series data in large-scale systems. Otus [24] collects and correlates performance metrics from distributed components and provides views of the data as a time-series. Both LiveRAC and Otus support visualization of generic time-series data. Artemis supports visualization plugins that display domain-specific histograms and time-series data. Theia, on the other hand, incoporates semantic information about job execution—allowing us to distinguish application-level problems from infrastructural problems.

Chau et al. [6] present an initiative for combining machine learning and visualizations for large network data exploration. Amershi et al. [1] use a similar approach for alarm triage on a network of more than 15,000 devices. We are investigating whether we can incorporate more sophisticated anomaly detectors into our tool.

6 Conclusion

Theia is a visualization tool that exploits application-specific semantics about the structure of MapReduce jobs to generate high-density, interactive visualizations of job performance that scale to support current industry deployments. Theia uses heuristics to identify visual signatures of problems that allow users to distinguish application-level problems (e.g., software bugs, workload imbalances) from infrastructural problems (e.g., contention problems, hardware problems). We have evaluated our visualizations using real problems experienced by Hadoop users at a production cluster over a one-month period. Our visualizations correctly identified 192 out of 204 problems that we observed.

In the future, we plan to integrate these visualizations as part of the standard stack of tools for Hadoop diagnosis. Detecting anomalies in heterogeneous clusters is an open issue as our assumption of peer-similarity might not hold in these systems. Other enhancements to the visual interfaces that we are considering are: 1) to display nodes hosted in the same rack together to increase our ability to spot failures at the rack-level; 2) to generate high-density per-task views and to incorporate resource-usage information such as CPU and memory usage; and 3) to develop algorithms that automatically classify application-level and infrastructural problems using the heuristics described in this paper.

7 Acknowledgments

We thank Mitch Franzos, Michael Stroucken, Zisimos Economou, and Kai Ren for access to the OpenCloud Hadoop logs. We thank our shepherd, Cory Lueninghoener, the conference reviewers, and our colleagues, Raja Sambasivan and Ilari Shafer, for their valuable feedback. We thank the members and companies of the PDL Consortium (including Actifio, APC, EMC, Emulex, Facebook, Fusion-io, Google, Hewlett-Packard Labs, Hitachi, Intel, Microsoft Research, NEC Labs, NetApp, Oracle, Panasas, Riverbed, Samsung, Seagate, STEC, Symantec, VMware, and Western Digital) for their support. This research was sponsored in part by the project CMUPT/RNQ/0015/2009 (TRONE—Trustworthy and Resilient Operations in a Network Environment), and by Intel via the Intel Science and Technology Center for Cloud Computing (ISTC-CC).

References

[1] AMERSHI, S., LEE, B., KAPOOR, A., MAHAJAN, R., AND CHRISTIAN, B. CueT: human-guided fast and accurate network alarm triage. In *ACM Conference on Human Factors in Computing Systems (CHI)* (Vancouver, BC, Canada, May 2011).

[2] ANANTHANARAYANAN, G., KANDULA, S., GREENBERG, A., STOICA, I., LU, Y., SAHA, B., AND HARRIS, E. Reining in the

outliers in map-reduce clusters using Mantri. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)* (Vancouver, BC, Canada, October 2010), pp. 1–16.

[3] BODIK, P., FRIEDMAN, G., BIEWALD, L., LEVINE, H., CANDEA, G., PATEL, K., TOLLE, G., HUI, J., FOX, A., JORDAN, M. I., AND ET AL. Combining visualization and statistical analysis to improve operator confidence and efficiency for failure detection and localization. In *IEEE International Conference on Automatic Computing (ICAC)* (Seattle, WA, June 2005), pp. 89–100.

[4] BOSTOCK, M., OGIEVETSKY, V., AND HEER, J. D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)* (2011).

[5] CAMPBELL, J. D., GANESAN, A. B., GOTOW, B., KAVULYA, S. P., MULHOLLAND, J., NARASIMHAN, P., RAMASUBRAMANIAN, S., SHUSTER, M., AND TAN, J. Understanding and improving the diagnostic workflow of mapreduce users. In *ACM Symposium on Computer Human Interaction for Management of Information Technology (CHIMIT)* (Boston, Massachusetts, December 2011).

[6] CHAU, D. H., KITTUR, A., HONG, J. I., AND FALOUTSOS, C. Apolo: making sense of large network data by combining rich user interaction and machine learning. In *ACM Conference on Human Factors in Computing Systems (CHI)* (Vancouver, BC, Canada, May 2011).

[7] CRETU-CIOCARLIE, G. F., BUDIU, M., AND GOLDSZMIDT, M. Hunting for problems with artemis. In *USENIX Workshop on Analysis of System Logs* (San Diego, CA, December 2008).

[8] DAI, J., HUANG, J., HUANG, S., HUANG, B., AND LIU, Y. Hitune: dataflow-based performance analysis for big data cloud. In *USENIX Annual Technical Conference (ATC)* (Portland, OR, June 2011).

[9] DEAN, J., AND GHEMAWAT, S. MapReduce: simplified data processing on large clusters. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)* (San Francisco, CA, December 2004), pp. 137–150.

[10] FONSECA, R., PORTER, G., KATZ, R. H., SHENKER, S., AND STOICA, I. X-Trace: A pervasive network tracing framework. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)* (Cambridge, MA, Apr. 2007).

[11] FORD, D., LABELLE, F., POPOVICI, F. I., STOKELY, M., TRUONG, V.-A., BARROSO, L., GRIMES, C., AND QUINLAN, S. Availability in globally distributed storage systems. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)* (Vancouver, CA, October 2010), pp. 61–74.

[12] KALAWSKY, R. S. Gaining greater insight through interactive visualization: A human factors perspective. In *Trends in Interactive Visualization*, R. Liere, T. Adriaansen, and E. Zudilova-Seinstra, Eds., Advanced Information and Knowledge Processing. Springer London, 2009.

[13] KANDULA, S., MAHAJAN, R., VERKAIK, P., AGARWAL, S., PADHYE, J., AND BAHL, P. Detailed diagnosis in enterprise networks. In *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)* (Barcelona, Spain, August 2009), pp. 243–254.

[14] KAVULYA, S., DANIELS, S., JOSHI, K. R., HILTUNEN, M. A., GANDHI, R., AND NARASIMHAN, P. Practical experiences with chronics discovery in large telecommunications systems. In *IEEE Conference on Dependable Systems and Networks (DSN)* (Boston, MA, June 2012).

[15] LIU, Z., LEE, B., KANDULA, S., AND MAHAJAN, R. Netclinic: Interactive visualization to enhance automated fault diagnosis in enterprise networks. In *IEEE Conference on Visual Analytics Science and Technology* (Salt Lake City, UT, October 2010), pp. 131–138.

[16] MAHIMKAR, A. A., GE, Z., SHAIKH, A., WANG, J., YATES, J., ZHANG, Y., AND ZHAO, Q. Towards automated performance diagnosis in a large IPTV network. In *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)* (Barcelona, Spain, August 2009), pp. 231–242.

[17] MASSIE, M. L., CHUN, B. N., AND CULLER, D. E. The ganglia distributed monitoring system: Design, implementation, and experience. *Parallel Computing 30* (June 2004), 817–840.

[18] MCLACHLAN, P., MUNZNER, T., KOUTSOFIOS, E., AND NORTH, S. C. LiveRAC: interactive visual exploration of system management time-series data. In *ACM Conference on Human Factors in Computing Systems (CHI)* (Florence, Italy, April 2008), pp. 1483–1492.

[19] NAGIOS ENTERPRISES LLC. Nagios. http://www.nagios.org/, September 2012.

[20] OLINER, A. J., KULKARNI, A. V., AND AIKEN, A. Using correlated surprise to infer shared influence. In *IEEE Conference on Dependable Systems and Networks (DSN)* (Chicago, IL, July 2010), pp. 191–200.

[21] PAN, X., TAN, J., KAVULYA, S., GANDHI, R., AND NARASIMHAN, P. Ganesha: Blackbox diagnosis of MapReduce systems. *SIGMETRICS Perform. Eval. Rev. 37* (January 2010).

[22] PARALLEL DATA LAB. Apache Hadoop. `http://wiki.pdl.cmu.edu/opencloudwiki/`, September 2012.

[23] RABKIN, A., AND KATZ, R. Chukwa: a system for reliable large-scale log collection. In *USENIX Conference on Large Installation System Administration (LISA)* (San Jose, CA, November 2010), USENIX Association.

[24] REN, K., LÓPEZ, J., AND GIBSON, G. Otus: resource attribution in data-intensive clusters. In *Workshop on MapReduce and its applications (MapReduce)* (San Jose, CA, June 2011).

[25] SAMBASIVAN, R. R., ZHENG, A. X., ROSA, M. D., KREVAT, E., WHITMAN, S., STROUCKEN, M., WANG, W., XU, L., AND GANGER, G. R. Diagnosing performance changes by comparing request flows. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)* (Boston, MA, March 2011), pp. 43–56.

[26] SIGELMAN, B. H., BARROSO, L. A., BURROWS, M., STEPHENSON, P., PLAKAL, M., BEAVER, D., JASPAN, S., AND SHANBHAG, C. Dapper, a large-scale distributed systems tracing infrastructure. Tech. rep., Google, Inc., 2010.

[27] TAN, J., KAVULYA, S., GANDHI, R., AND NARASIMHAN, P. Visual, log-based causal tracing for performance debugging of mapreduce systems. In *IEEE International Conference on Distributed Computing Systems (ICDCS)* (Genova, Italy, June 2010), pp. 795–806.

[28] TAN, J., PAN, X., KAVULYA, S., GANDHI, R., AND NARASIMHAN, P. SALSA: Analyzing Logs as StAte Machines. In *USENIX Workshop on Analysis of system logs* (San Diego, California, December 2008).

[29] THE APACHE SOFTWARE FOUNDATION. Apache Hadoop. `http://hadoop.apache.org/`, September 2012.

[30] THE APACHE SOFTWARE FOUNDATION. PoweredBy Hadoop. `http://wiki.apache.org/hadoop/PoweredBy`, September 2012.

[31] THUSOO, A., SHAO, Z., ANTHONY, S., BORTHAKUR, D., JAIN, N., SEN SARMA, J., MURTHY, R., AND LIU, H. Data warehousing and analytics infrastructure at facebook. In *ACM Conference on Management of Data (SIGMOD)* (Indianapolis, IN, June 2010).

[32] TUFTE, E. R. *The Visual Display of Quantitative Information*, 2nd ed. Graphics Pr, May 2001.

[33] WARE, C. *Visual Thinking: for Design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.