3-2022

BatchLens: a visualization approach for analyzing batch jobs in cloud systems

Shaolun RUAN
Singapore Management University, slruan.2021@phdcs.smu.edu.sg

Yong WANG
Singapore Management University, yongwang@smu.edu.sg

Hailong JIANG

Weijia XU

Qiang. GUAN

Citation

BatchLens: A Visualization Approach for Analyzing Batch Jobs in Cloud Systems

Shaolun Ruan
School of Computing and Information Systems
Singapore Management University
Singapore, Singapore
slruan.2021@phdcs.smu.edu.sg

Yong Wang
School of Computing and Information Systems
Singapore Management University
Singapore, Singapore
yongwang@smu.edu.sg

Hailong Jiang
Department of Computer Science
Kent State University
Kent, U.S.
hjiang13@kent.edu

Weijia Xu
Scalable Computational Intelligence Group
Texas Advanced Computing Center
Austin, U.S.
xwj@tacc.utexas.edu

Qiang Guan
Department of Computer Science
Kent State University
Kent, U.S.
qguan@kent.edu

arXiv:2112.15300v1 [cs.HC] 31 Dec 2021

Abstract—**Cloud systems are becoming increasingly powerful and complex. It is highly challenging to identify anomalous execution behaviors and pinpoint problems by examining the overwhelming intermediate results/states in complex application workflows. Domain scientists urgently need a friendly and functional interface to understand the quality of the computing services and the performance of their applications in real time. To meet these needs, we explore data generated by job schedulers and investigate general performance metrics (e.g., utilization of CPU, memory and disk I/O). Specifically, we propose an interactive visual analytics approach, *BatchLens*, to provide both providers and users of cloud service with an intuitive and effective way to explore the status of system batch jobs and help them conduct root-cause analysis of anomalous behaviors in batch jobs. We demonstrate the effectiveness of *BatchLens* through a case study on the public Alibaba bench workload trace datasets.**

Index Terms—**cloud computing, visual analytics, human-computer interaction**

I. INTRODUCTION

Cloud computing has become the backbone of modern IT systems, which supports processing large volumetric data using clusters of computing nodes [1], [2]. Understanding the batch jobs' behaviors on cloud platforms is of great importance for cloud providers and cloud service users. Anomalous behaviors of batch jobs can potentially indicate existing software bugs and hardware crashes, which will eventually result in the violation of the Service Level Agreement (SLA) [3]. However, it is still a challenging and complex task to diagnose and prevent anomalous execution behaviors in cloud computing environments [4].

To prevent software flaws and hardware accidents that can result in the failure of cloud services, cloud providers have been monitoring cloud platforms through metrics-based [5], [6], log-based [7] and trace-based [8], [9] approaches. More recently, deep learning-based approaches are also used for anomalous behavior detection [10]. Prior studies, though, are effective techniques to monitor job and system behaviors, the cause is still invisible to the cloud system administrators due to the hidden patterns of the batch job co-allocation. Meanwhile, the existing tools are generally designed for system administrators, users may also need to monitor the status of their executing jobs so that they can provide more detailed information to system administrators when submitting the tickets. Moreover, the preceding methods are neither intuitive nor efficient as they consist of large-scale general metric data, which significantly affects the perception of the abnormal status of compute nodes and makes system administrators suffer from monitoring inflexibility.

Visualization tools have been extensively adopted to support offline log analysis in a variety of cloud applications. Prior studies [11] have demonstrated that visual representation can provide rich insights in monitoring cloud computing performance, and increase the possibility of uncovering hidden patterns in cloud infrastructures through intuitive visual representations and effective user interactions.

In this paper, we propose a visualization approach *BatchLens* to analyze and monitor the job execution behaviors in cloud computing systems. Compared with existing works, *BatchLens* leverages effective visual representations and flexible interactions to analyze and detect anomalous batch jobs in cloud systems. Using the traces from large-scale parallel cloud systems at Alibaba, we develop multiple mutually-linked views to analyze the running jobs on metric-heavy compute nodes and enhance the effective human perception for the batch jobs. Specifically, interactive visual designs of hierarchical bubble charts and line charts are proposed to support analysis of abnormal jobs through temporal and spatial comparison. The major contributions of this paper can be summarized as follows:

- We propose a novel visualization approach *BatchLens* based on batch hierarchy data to enable interactive analysis of the batch jobs in cloud systems.
- We conduct a case study on the public Alibaba trace

datasets to demonstrate the effectiveness of our proposed approach.

II. DATASET AND DAG BATCH WORKLOADS

Alibaba trace datasets [12] is part of the Alibaba Open Cluster Trace Program, which contains performance profile data collected from the Alibaba large-scale distributed cloud computing platform across 1300 machine batch jobs and a 24-hour duration. In this paper, we only focus on batch jobs and their dependencies. Each trace record in batch scheduler data includes the hierarchical structure for a compute node set at a 300-second resolution. For server usage data set, each row includes metadata of the node and the performance log of three metrics, i.e., GPU utilization, memory utilization and disk utilization at a one-second resolution. A task has one or multiple instances running on the respective compute node. According to our data pre-processing, 75% batch jobs contain only one task, while 94% tasks have multiple instances. Note that each instance must be executed by only one compute node, and each compute node can run multiple instances simultaneously.

III. VISUAL DESIGN

To analyze the batch job status from the perspectives of both their spatial distribution and temporal evolution, *BatchLens* provides users with multiple linked views to reveal the insights of batch job scheduling (Fig. 3). Also, rich interactions are enabled to facilitate convenient explorations.

A. Hierarchical Bubble Chart

The hierarchical bubble chart is an overview of batch dependency, which provides a comprehensive hierarchy of batch jobs, tasks and instances (corresponding compute nodes). We adopt hierarchical bubble charts, as it can intuitively visualize the hierarchy of multiple nodes.

Specifically, three layers of bubbles are applied to indicate the hierarchical batch entities (Fig. 1). Bubbles highlighted with blue dotted circles denote batch job level, which contains the child level of tasks highlighted by purple dotted circles. Each compute node, which is scheduled to execute the batch instances and is subordinated to the respective task, is comprised of three parts denoting general usage metrics, i.e., CPU utilization, memory utilization and disk I/O utilization. We colorize the state metrics to reflect the performance of the machines at once.

B. Line Charts

Temporal analysis on cloud computing systems facilitates the detection of anomalous performances of compute nodes over time. We utilize line charts to reflect metric trends and incorporate multiple vertical annotation lines into line charts for start time and end time representation of batch jobs.

As shown in Fig. 2, line charts are used to indicate changes of machine utilization over time. Specifically, it shows metric trends of those compute nodes executing the same batch job simultaneously. For example, Fig. 2(a) visualizes the CPU utilization of all the nodes executing *job_7399* in the overall time period. Green annotation lines denote the start time of job
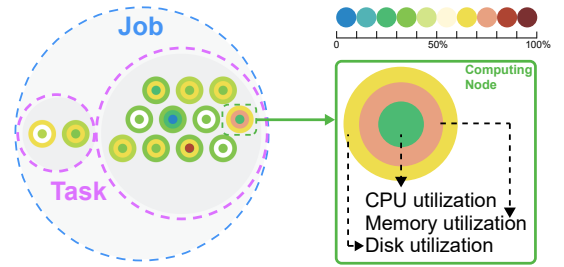

Fig. 1. Visualization of batch scheduling data encoded by hierarchical bubbles and the color scheme for performance metrics, i.e., CPU utilization, memory utilization and disk utilization, which are indicated by three annuli in the detailed view.
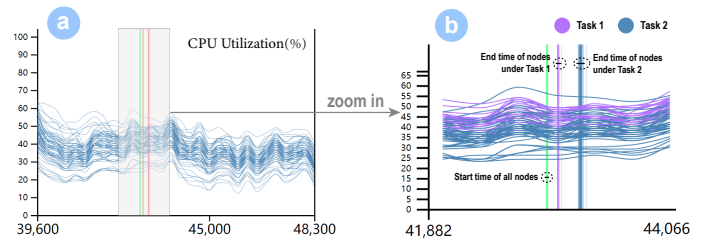

Fig. 2. Multiple line chart reflecting the metric utilization changes of nodes under a selected batch job over time. (a) indicates the utilization trend in the overall time period. After selecting the time range via brushing, (b) is generated to show the detailed view of the selected part. For both views, vertical annotation lines in green and non-green are used to show the start time and end time of the job execution respectively.

execution on corresponding nodes. All lines bundling into one cluster indicates that the job is scheduled for all nodes at the same time. Red lines depict end timestamps of job execution, which are bundled as two clusters, as *job_7339* includes two tasks and each has a different end timestamp. Also, after selecting the interesting time range of overall line charts by brushing, users can explore the detailed metric utilization (Fig. 2(b)). In the selected detailed view, different lines and annotation lines are plotted in various colors, which enables users to compare node usages by task dimension. By interacting with the line charts of various batch jobs, users can observe the temporal patterns in terms of metric trends of compute nodes, such as a spike or a valley in the context of other nodes' performance.

C. User Interactions

Flexible and intuitive interactions allow a comprehensive analysis of the batch job observation. A simple timeline is used to represent the metrics aggregated across the entire cloud systems over time. Each layer of the graph represents one metric. Users can select an interesting time range and timestamp through the brushing and choosing interaction respectively.

Also, another simple user interaction is adopted to recognize the same compute node executing various batch jobs simultaneously. Since the hierarchical bubble chart is a job-based graph, the same node may be rendered into multiple parent job bubbles. A direct mouse over on target nodes will trigger a zoom-in refresh, as shown in pairwise nodes that are linked with the same color dotted lines in Fig. 3(b).
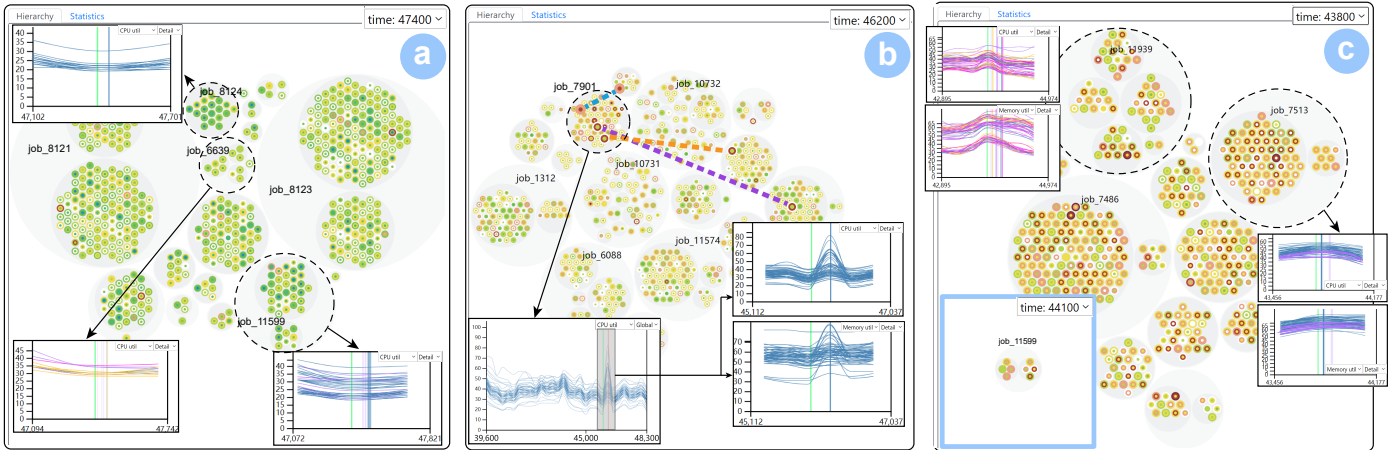
Fig. 3. The main view and corresponding detailed views of the visual analytics system. Detailed views of multiple line charts show the metric utilization trends of compute nodes under a selected batch job over time. The green annotation lines in each line chart depict the start timestamps of the selected job across each compute node executing it, while other annotation lines denote the job end timestamps using the same color as the corresponding lines of nodes.

IV. CASE STUDY

We conduct a case study using Alibaba trace datasets to demonstrate the effectiveness of the proposed visualization approach. We select three interesting and representative timestamps to illustrate the hidden patterns of abnormal machine status (Fig. 3). It is clear that both figures are uniform in color distribution due to the load balance.

Fig. 3(a) shows a common situation of the system, where all the machines that host some tasks are at low resource utilization (20% - 40%), and all the performance metrics are stable. Specifically, from the hierarchical bubble chart, we can observe that there are 15 root bubbles denoting batch jobs at timestamp 47400, which includes two primary jobs (Job *job_8121* and Job *job_8123*), both of which are scheduled into two tasks that are executed a substantial volume of nodes. More specifically, Job *job_8124*, which is scheduled into one task only, has the nodes with the lowest utilization (CPU, memory and disk I/O). From the line chart corresponding to Job *job_8124*, we can see that the CPU utilization of all nodes is fairly constant with only small increase during the period of job execution (between the creation and termination of job). Also, for Job *job_6639*, the lines denoting CPU utilization of nodes under four separated tasks are plotted with four colors. These four types of tasks, as the annotation lines plots, have the same start timestamp but multiple end timestamps. However, CPU utilization for all the nodes in different tasks stays stable during job executions. The same pattern can be observed for Job *job_11599*.

In bubble chart of Fig. 3(b), it is clear that all nodes are running at medium level of resource utilization around Timestamp 46200 (50%-80%). The resource utilization on the nodes hosting the jobs are heavier than that in Fig. 3(a) through the color distribution of bubble charts, with an exception of Job *job_7901* running on busier nodes than those hosting other jobs. From the line chart of the overall time period of job of *job_7901* (left bottom view), we can see that the CPU utilization of corresponding nodes is synchronized, even though drastic fluc-

tuations exist. From the detailed view (right bottom view), a notable spike emerges for CPU and memory usage after Job *job_7901* is scheduled into the corresponding machines. Both metrics reach the peak of the utilization when the job execution is over, followed by a slow drop to the normal level, which indicates that the machines running Job *job_7901* experience intensive workload during the execution time. Additionally, we connect the same machines with colored dotted lines (green, orange and purple) in the bubble chart to help trace down the machines execute multiple tasks simultaneously.

More interesting findings can be revealed from Fig. 3(c). A tremendous amount of nodes are running at high CPU- or memory-utilization at Timestamp 43800, including several nodes reaching the respective capacity of node performance. From the line charts generated from Job *job_7513* at the bottom right, CPU utilization of nodes under two tasks is distinguished by blue and purple lines. Line cluster in purple depicts the relatively smaller task set, which has a less severe CPU and memory usage. Also, for Job *job_11939*, lines denoting five different tasks are entangled seriously as shown in the detailed view on the top left. The same pattern can be perceived between views for CPU and memory utilization: an obvious drop occurs after the creation of Job *job_11939*. Moreover, we find that at Timestamp 44100, all of the preceding nodes on the system are shut down (figure at bottom left in Fig. 3(c)), and only Job *job_11599* is left on the entire platform. However, the general metrics still exist for the corresponding machines at Timestamp 44100 in the detailed line chart on the top left. It is likely to speculate that the compute node is suffering thrashing while the virtual memory is overused with the degree from multi-programming increasing. Eventually thrashing forces the CPU utilization to decrease and the whole system is not making any progresses. From the observation in the next time slice when almost all jobs disappear, these jobs are very likely terminated and relaunched by the user or system administrator to clear the thrashing.

V. Related Work

This work is related to prior research on anomalous behavior detection in cloud systems and visualization for cloud computing analysis.

A. Anomalous Behavior Detection in Cloud Systems

Anomalous behavior detection of the distributed system is an essential and challenging topic which attracts great research attention. Prior works can be categorized into three groups, i.e., metric-based, log-based and trace-based approaches. Metric-based [13] approach usually applies statistics on collected metrics, which mainly include performance metrics, e.g., throughput and system metrics, e.g., CPU, memory and I/O. Logs are generated along with applications' running and reflect application processing status and execution logic. Recent studies [14], [15] leverage logs or traces to create workflow models for software testing and the understanding of system behaviors. Lou et al. [16] proposed an automaton model for reconstructing concurrent workflows from event traces. Traces record information for program debugging or diagnosis purposes. The preceding studies are not preferred as without specifically-designed visualization methods, inflexible row cloud trace data can not be presented by intuitive visual summarization providing quick and efficient analytic process.

B. Visualization for Cloud Computing Analysis

Existing visual metaphors [17], [18] have studied representations on comparison of quantitative state changes (e.g., hardware metrics of compute nodes), while they are not suitable for our application scenario as the trace data includes spatial characteristics for topological batch distribution. Many visualization tools [19], [20] serve as collecting low-level trace data from large parallel systems. More recently, Muelder et al. [21] proposed a typical system for cloud computing analysis to portray the behavior of each compute node over time. A variety of visualizations [22] have been proposed for monitoring and analyzing trace data generated from cloud computing systems. Though the preceding studies provide rich insights of large-scale parallel network analysis, they rarely analyze the anomalous batch jobs, which supports the anomalous behavior detection and further conducts root-cause analysis.

VI. Conclusion and Future Work

We propose a visualization approach to interactively analyze the execution behavior using general performance metrics from Alibaba trace datasets. We have demonstrated the effectiveness with case studies and revealed three existing patterns of batch jobs on cloud systems, all of which can support the perception of the hidden reasons behind hardware-heavy compute nodes. Although our technique may not present every facet of the reasons for the anomalies, it provides system administrators and cloud users with deep insights into batch job status, and facilitate an easy detection of anomalies.

In future work, we would like to further extend our approach from two directions. First, real-time techniques have been extensively adopted on large-scale cloud computing platforms. We plan to extend *BatchLens* into a real-time online system and integrate it into real cloud distributed systems. Furthermore, *BatchLens* is an effective approach to detect abnormal jobs through visualizing their hardware performance metrics. But some hidden anomalies will not affect hardware performance significantly due to the load balance. It will be interesting to further investigate how to recognize those hidden abnormal statuses.

References

[1] Rimal, Bhaskar Prasad, Eunmi Choi, and Ian Lumb. "A taxonomy and survey of cloud computing systems." 2009 Fifth International Joint Conference on INC, IMS and IDC. Ieee, 2009.

[2] Lu, Gang, and Wen Hua Zeng. "Cloud computing survey." Applied Mechanics and Materials. Vol. 530. Trans Tech Publications Ltd, 2014.

[3] Meng, Fan Jing, et al."Driftinsight: detecting anomalous behaviors in large-scale cloud platform" 2017 IEEE 10th International Conference on Cloud Computing (CLOUD). IEEE, 2017.

[4] Dean, Daniel J., et al. "Automatic server hang bug diagnosis: Feasible reality or pipe dream?." 2015 IEEE International Conference on Autonomic Computing. IEEE, 2015.

[5] Ng, Fred. "Forcast: Public Cloud Services, Worldwide, 2014-202, 4Q16 Update." Gartner Inc. Gartner Report G 320866.

[6] Gu, Xiaohui, and Haixun Wang. "Online anomaly prediction for robust cluster systems." 2009 IEEE 25th International Conference on Data Engineering. IEEE, 2009.

[7] Fu, Qiang, et al. "Execution anomaly detection in distributed systems through unstructured log analysis." 2009 ninth IEEE international conference on data mining. IEEE, 2009.

[8] Zhang, Hui, et al. "CLUE: System trace analytics for cloud service performance diagnosis." 2014 IEEE Network Operations and Management Symposium (NOMS). IEEE, 2014.

[9] Dean, Daniel J., et al. "Perfscope: Practical online server performance bug inference in production cloud computing infrastructures." Proceedings of the ACM Symposium on Cloud Computing. 2014.

[10] Dean, Daniel Joseph, Hiep Nguyen, and Xiaohui Gu. "Ubl: Unsupervised behavior learning for predicting performance anomalies in virtualized cloud systems." Proceedings of the 9th international conference on Autonomic computing. 2012.

[11] Xu, Ke, et al. "Clouddet: Interactive visual analysis of anomalous performances in cloud computing systems." IEEE transactions on visualization and computer graphics 26.1 (2019): 1107-1117.

[12] Alibaba. 2017. Alibaba Open Cluster Trace Program. https://github.com/guanxyz/clusterdata/tree/master/cluster-trace-v2017

[13] Guan, Qiang, and Song Fu. "Adaptive anomaly identification by exploring metric subspace in cloud computing infrastructures." 2013 IEEE 32nd International Symposium on Reliable Distributed Systems. IEEE, 2013.

[14] Beschastnikh, Ivan, et al. "Mining temporal invariants from partially ordered logs." Managing Large-scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques. 2011. 1-10.

[15] Lo, David, Leonardo Mariani, and Mauro Pezzè. "Automatic steering of behavioral model inference." Proceedings of the 7th joint meeting Of The European Software Engineering Conference and the ACM SIGSOFT symposium on The foundations of software engineering. 2009.

[16] Lou, Jian-Guang, et al. "Mining program workflow from interleaved traces." Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. 2010.

[17] Tufte, Edward. "The visual display of quantitative information." (2001).

[18] Ruan, S., Wang, Y., & Guan, Q. (2021). "Intercept Graph: An Interactive Radial Visualization for Comparison of State Changes." In Proceedings of IEEE VIS 2021 (Short Paper).

[19] Shende, Sameer S., and Allen D. Malony. "The TAU parallel performance system." The International Journal of High Performance Computing Applications 20.2 (2006): 287-311.

[20] Zaki, Omer, et al. "Toward scalable performance visualization with Jumpshot." The International Journal of High Performance Computing Applications 13.3 (1999): 277-288.

[21] Muelder, Chris, et al. "Visual analysis of cloud computing performance using behavioral lines." IEEE transactions on visualization and computer graphics 22.6 (2016): 1694-1704.

[22] Sigovan, Carmen, Chris W. Muelder, and Kwan-Liu Ma. "Visualizing Large-scale Parallel Communication Traces Using a Particle Animation Technique." Computer Graphics Forum. Vol. 32.