JobDigest – Detailed System Monitoring-Based Supercomputer Application Behavior Analysis

Dmitry Nikitenko ✉ [0000-0002-2864-7995], Alexander Antonov [0000-0003-2820-7196], Pavel Shvets [0000-0001-9431-9490], Sergey Sobolev [0000-0002-3474-8696], Konstantin Stefanov [0000-0002-0930-2713], Vadim Voevodin [0000-0003-1897-1828], Vladimir Voevodin [0000-0001-6036-5106], and Sergey Zhumatiy [0000-0001-5770-3071]

Research Computing Center, M.V. Lomonosov Moscow State University, Moscow, 119234, Russian Federation
{dan, asa, shpavel, sergeys, cstef, vadim, voevodin, serg}@parallel.ru

Abstract. The efficiency of computing resources utilization by user applications can be analyzed in various ways. The JobDigest approach based on system monitoring was developed in Moscow State University and is currently used in everyday practice of the largest Russian supercomputing center of Moscow State University. The approach features application behavior analysis for every job run on HPC system providing: the set of dynamic application characteristics - time series of values representing utilization of CPU, memory, network, storage, etc. with diagrams and heat maps; the integral characteristics representing average utilization rates; job tagging and categorization with means of informing system administrators and managers on suspicious or abnormal applications. The paper describes the approach principles and workflow, it also demonstrates JobDigest use cases and positioning of the proposed techniques in the set of tools and methods that are used in the MSU HPC Center to ensure its 24/7 efficient and productive functioning.

Keywords: HPC · Supercomputing · Efficient Computing · Resource Utilization · Job Dynamics · Application Efficiency · Parallel Programming.

1 Introduction

The JobDigest[1] approach follows monitoring of HPC application performance principles – one of the possible ways of resource utilization efficiency analysis for both applications and supercomputers. Studying of the resource utilization type and rate, determining bottlenecks in programs, hardware and/or their interplay are the typical usage scenarios for such methods. The characteristics of the HPC system components state serve the basis for these approaches. For example, various CPU load types, incoming and outgoing network traffic on the node, number of floating

[1] The JobDigest® is a registered trademark in Russian Federation. The application for an invention of the JobDigest approach was filed.

point or integer operations, accelerator usage, memory-related operations like number of load and store operations, misses in the cache memory of different levels and so on, input/output activity, and many other characteristics.

Most of available monitoring systems do not process the data at computing nodes to reduce the impact on job execution [1-7]. At the same time there are uprising monitoring systems that reasonably distribute the processing of data and secure low influence on job execution [8, 9]. The key advantage of monitoring-based approaches is general independence on the code, absence of necessity to make any changes to the program source code or binary that is being analyzed. This allows reducing the influence on program execution and at the same time expanding the range of jobs that can be analyzed.

Different approaches to job performance monitoring are known [10-17]. In this paper the details of the JobDigest approach to supercomputer job analysis based on system monitoring data is described. According to the approach principles, the special report that gives all-round view over the job behavior built of diverse dynamic and integral characteristics is generated for every job, even if it has not successfully finished.

2 Approach Principles

The JobDigest report represents the detailed information of job behavior starting from basic job information to the precise info on computing, storage and network resource utilization.

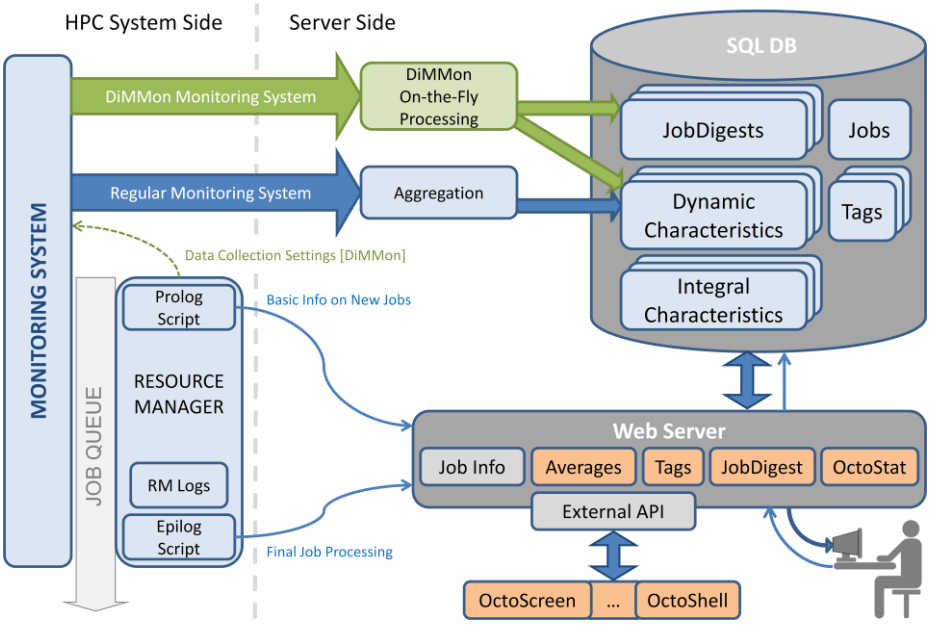

Fig. 1. General proposed workflow based on system monitoring data analysis.

The key feature of the proposed approach is the possibility of job behavior analysis for any and every run with no code preparations by the user [18, 19].

The JobDigest is built on the basis of system monitoring data and job details from the resource manager [20]. As soon as the job is assigned to the set of nodes, it becomes possible to bind the system monitoring data collected from the corresponding nodes to form the profile of application execution (Figure 1).

It is supposed that only one job at a time is allocated to the node. If a node can be assigned to several jobs, such a situation can also be handled to some extent but only if process to core pinning is enabled.

3 The JobDigest Report

The report consists of several blocks that focus on various scopes of analysis. Altogether the blocks shape all-round basis for application behavior study (Figure 2).

A. General job information. The table includes the general information on job and its node allocation from the resource manager: job ID, user account, run command, working directory, output file, partition, time limit, submit/start/end timestamps, status, duration, number of allocated cores and nodes, core hours, list of allocated nodes.

B. Dynamical job characteristics. Dynamic characteristics represent the rate of corresponding resource utilization during the program execution. Originally the data granularity can be rather high, up to 10 Hz, but for the most jobs it is not necessary to have such detailed information, keeping in mind large volumes of data to be stored in this case. For the most cases, granularity reduced to 5 minutes is enough to allow having clear overall view of the application behavior. The DiMMon monitoring system will allow dynamical reconfiguration of the granularity for selected jobs, partitions, and so on.

Every dynamic characteristic is represented by five values for every time interval: min, max, min_avg, max_avg, avg. With present settings node_min, node_max and node_avg sensor values are aggregated for every 5 minute time interval from each node. Based on these three values, min(node_min), max(node_max) и avg(node_min), avg(node_max), avg(node_avg) are calculated across all job nodes leading to five values mentioned earlier.

The available set of dynamic characteristics can vary depending on analysis purposes and system settings. There are over 20 different characteristics available for «Lomonosov» and «Lomonosov-2» systems at present: CPU user load, load average, free memory, L1 cache misses, L2 cache misses, last level cache misses, MPI IB receive data, MPI IB send data, MPI IB receive packets, MPI IB send packets, FS IB receive data, FS IB send data, FS IB receive packets, FS IB send packets, instructions retired, memory load, memory store, CPU nice load, CPU system load, CPU idle, CPU IO wait load, CPU IRQ load, CPU soft IRQ load, etc.

There are three ways of studying dynamic characteristics supported by JobDigest at present: diagrams, CSV tables for using with external analysis and visualization

tools, and heat maps. Heat maps are 2D charts. Horizontal axis corresponds to time, and vertical axis corresponds to used nodes. The dot color represents dynamic characteristic value from minimum (red color) to maximum (green color) among all the values during the job execution. An example of such heat map is shown in Figure 8.

Fig. 2. JobDigest report blocks: A – general job information, B – dynamic job characteristics, C – integral job characteristics, D – tags and job categories.

C. Integral job characteristics. The integral job characteristics represent average resource utilization and are built on the base of dynamic job characteristics. Every integral characteristic is provided as a set of minimum, average, and maximum levels during the whole job run. Every of these three values can be highlighted according to preset and calculated thresholds.

In the same block there are also given some other characteristics built as derivatives of averages. For example, the following average characteristics can be useful and are available in the report: IB receive packet size for FS, IB receive packet size for MPI, IB send packet size for FS, IB send packet size for MPI, and some memory-related characteristics like level 1 to level 3 cache miss ratio and memory load plus memory store to level 1 cache miss ratio.

D. Job categories and tags. Based on calculated integral characteristics and general job information every job is tagged after it is finished. The tags help to divide jobs into categories, helping to find specific jobs later by category or categories intersection. The tags mark the scale of the job, partition, duration, resource utilization specifics, etc.

The examples of the report and its blocks are provided in the "Evaluation and Use Cases" section of the paper.

4 Complementarity with Other HPC Tools

JobDigest interaction with other components of the toolkit mentioned in Figure 1 and Figure 3 is widely used every day in the MSU HPC Center [21]. All these components have been developed in the Research Computing Center of Moscow State University.

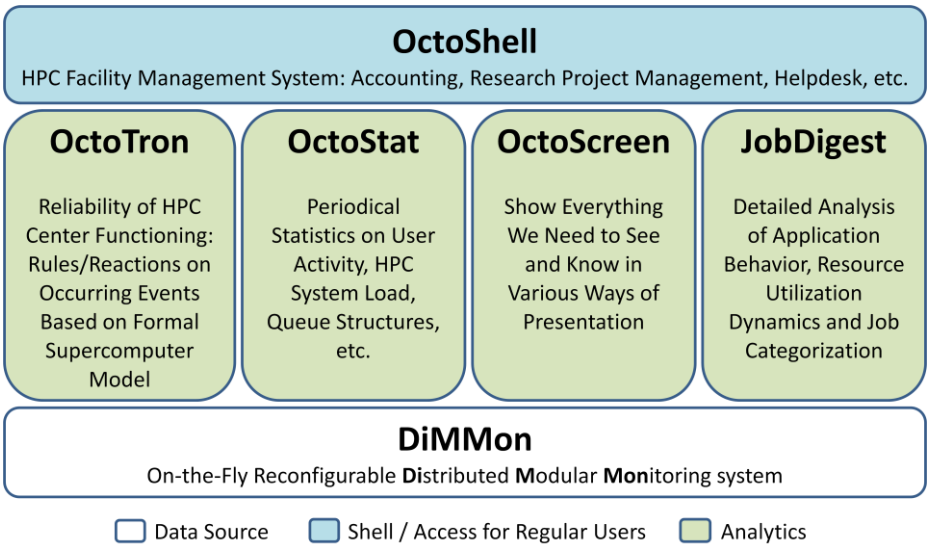

Fig. 3. Main MSU toolkit components for HPC centers.

DiMMon. This monitoring system is a promising scalable reconfigurable data collector with elements of on-the-fly analysis. In comparison to other monitoring systems, it allows performing much of data processing before saving source data to the database. For example, it allows creating JobDigest reports and calculating averages for the running jobs.

OctoScreen. Administrators and system managers can proceed to the more detailed job JobDigest reports from specifically generated job lists provided by OctoScreen. Such lists can be formed by various criteria, for example, by job owner, responsible organization for the research project that the jobs belong to, geographical criteria, research areas and so on [22].

OctoStat. Provided by Octostat analysis of daily statistics on queue structure and top resource consuming projects, users and accounts can be successfully amplified having the possibility to look at the details of any job owned by a specific account, user or research project. This is especially valuable for suspicious jobs that are found according to extremely low activity or suspicious node allocation.

OctoTron. The resilience system logs all problems with storage, network, compute nodes and the interfaces [23]. If something goes wrong, it is possible to track the jobs that could have been harmed by the issues with the known node set in the specified time period and create the list of potentially affected job runs. One can see more details to find out if there was really something wrong with the job proceeding to the JobDigest reports of any of these jobs.

OctoShell. If the HPC system is not a dedicated one and there is a number of users and research projects that should not be allowed to see the working results and activity details of each other, it is essential to have a special system that would serve as a single entry point for all or, at least, most of services. The OctoShell system is used in the MSU HPC Center for such purposes [24, 25]. All the information on job runs is provided to users through this system according to user access permissions and settings in the account area on the website [26].

As shown above, JobDigest is highly integrated in the HPC center administration and management workflow as a valuable analytical part of the used toolkit.

5 Implementation Details

The workflow of the current version of JobDigest is presented on Figure 4. The numbers in circles present the sequence of stages.

1. System monitoring data from agents of monitoring system on the nodes of HPC system is sent to the aggregation service.
2. The aggregation service filters data and reduces granularity. Data is further saved into the database. It is not pinned to jobs yet.
3. As soon as job starts, prolog script of resource manager informs server-side application on the web server on job run. Server-side script writes the initial info into the database.

4. When the job finishes, epilog script informs server-side application on the web server on job end. Server-side script updates the job record in the database and initiates job data processing in background mode: integral characteristics are being calculated and written into the database. Job tagging is performed afterwards based on integral characteristics and general job information, the tags assignment is written into the database.
5. Special script gathers information on all run jobs and checks if all the jobs are represented in the database. This is done once per day to tackle possible problems with network during the job run and other issues that could have prevented the job info getting processed.
6. External services and applications can access job information and monitoring data, as well as daily statistics data in JSON format via HTTP protocol using special API. For example, regular users access the JobDigest reports and get job run statistics via OctoShell.
7. Administrators can access job info and corresponding monitoring data, job lists, extended JobDigest reports, statistics and special visualization templates as HTML pages by HTTP with authorization.

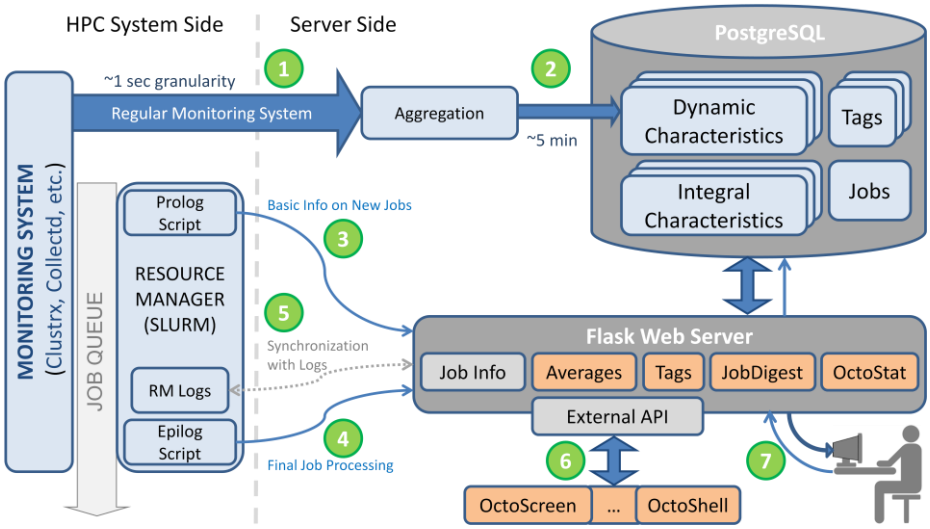

Fig. 4. JobDigest workflow stages.

At present the following software components are used:

– PostgreSQL, version 8.4.20;
– Flask web server, version 0.11.1;
– SLURM resource manager [27], «Lomonosov» HPC system – v.2.5.6, «Lomonosov – 2» system - v.15.0.8.1;
– Modified ClustrXWatch agents are used now as the data source [28];
– Google Charts are used for diagrams in the JobDigest reports;
– Highcharts are used for heat map generation.

The latest versions of the JobDigest components are available at GihHub [29]. As for now the custom installation is a bit tricky. As soon as the development of DiMMon monitoring system allows, the out-of-the-box package will be available.

6 Evaluation and Use Cases

The first approach evaluation was achieved as a result of MSU team efforts under the joint RU-EU HOPSA project [30].

The described approach is now widely used in the everyday practice of the Moscow State University Supercomputer Center, the largest HPC collaborative facility in Russia having «Lomonosov» and «Lomonosov-2» systems with a total of over 4 PFlops peak performance at present, over 500 collaborative research projects and thousands of scientists using the computing facility in 24/7 mode. This results in processing of thousands of supercomputer jobs per day [31], and the special JobDigest report generation is provided for any and every of those. In this section we illustrate some examples of using JobDigest report and its data.

6.1 Integral job characteristics

Every JobDigest report contains a block with integral job characteristics that represent general resource utilization rate for the whole run. It is typically MIN, MAX and AVG values for every dynamic characteristic available, see the description in subsection C of section 3 of the paper.

Lomonosov running tasks

id	account	partition	t_start	num_cores	avg cpu_user	avg loadavg	comment
[illegible]	[illegible]	regular4	2017-05-18 13:04:46	128	0.03	0.01	HANGED?!
[illegible]	[illegible]	gpu	2017-05-15 16:30:45	128	60.97	7.78	
[illegible]	[illegible]	regular6	2017-05-15 22:27:30	48	46.87	12.00	
[illegible]	[illegible]	regular6	2017-05-16 07:24:30	48	40.11	10.00	
[illegible]	[illegible]	regular6	2017-05-15 20:53:04	48	47.52	12.00	
[illegible]	[illegible]	regular6	2017-05-16 01:41:40	192	35.89	3.01	

Fig. 5. JobDigest list with suspicious jobs.

These integral characteristics are available not only from inside the report. Most of them are provided together with the job list according to access permissions of the user (own jobs for user, selected jobs for the experts, or all jobs for administrators). In many cases the general average resource utilization rate is already can be sufficient for preliminary analysis and/or job selection for detailed study.

Figure 5 shows the example of such a table with integral characteristics that can help to find the hanged jobs among the currently run jobs according to extremely low CPU_user.

The layout of such a job list can vary and can contain many integral characteristics at a time.

One of the possible promising ways of using integral characteristics is application scalability analysis that can be performed analyzing changing integral job characteristics of sequence of runs [32].

6.2 Job categories and tags

Integral characteristics values can serve as the basis for job categorization and tagging according to belonging to the specific job group or type [33]. For example (Figure 6), if no GPU usage is observed in GPU partition job, one can suspect a cheating - the regular CPU job could have been intentionally put to GPU partition just because of the shorter queue, blocking GPU resources for GPU oriented jobs.

Another promising way of job categorization and finding job anomalies bases on data mining principles and processing of historical dynamic job characteristics, the first results are described in [34].

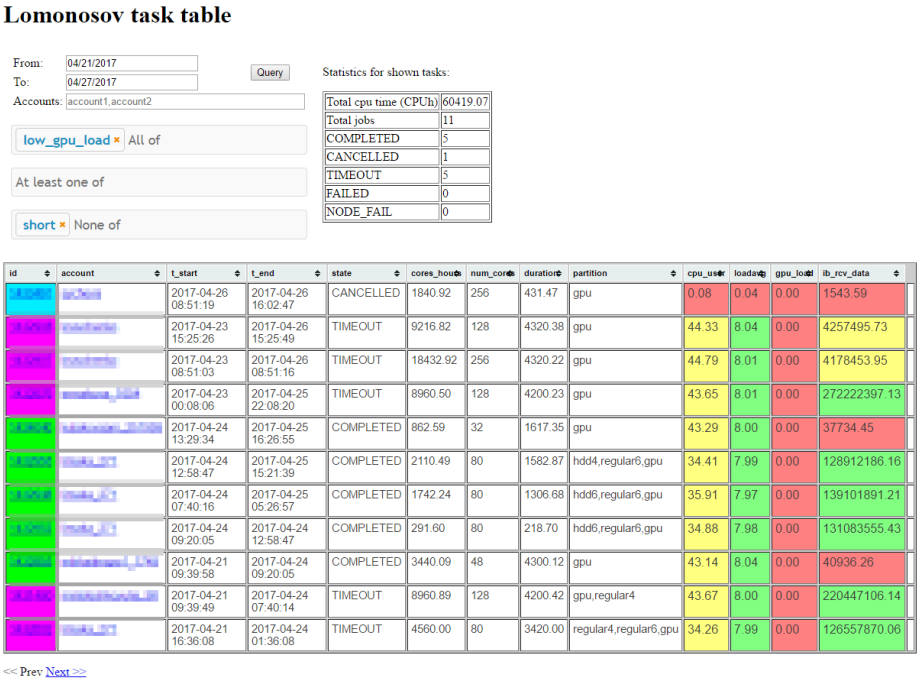

Fig. 6. JobDigest list of "low_gpu_load" tagged categories.

As shown above, the job lists can be equipped with basic filtration tools by time, account name, categories (tags), and can contain adjustable number of general and integral job characteristics. In JobDigest, the job categories and tags are shown as described in subsection D of section 3.

6.3 Dynamic job characteristics

The dynamics of execution and behavior of any job can be best observed by analyzing the behavior of dynamic characteristics and their interplay. There are three general modes of analysis available: CSV export for external tools, diagrams, and heat maps.

Figure 7 illustrates changing of dynamic job characteristics during the job execution with over 200 processes used. The blue color lines (upper) correspond to maximum values of all the processes, green lines correspond to averages. In the second part of job execution it can be seen that the average values of all characteristics become very low - almost drop to zero level. At the same time the maximum value still represents normal activity. Knowing that the number of processes of the job was about several hundred, such a behavior can be possibly explained by the activity of just a few processes while the rest hundreds of processes are in a wait state. In any case, such a behavior found using the JobDigest report is suspicious.

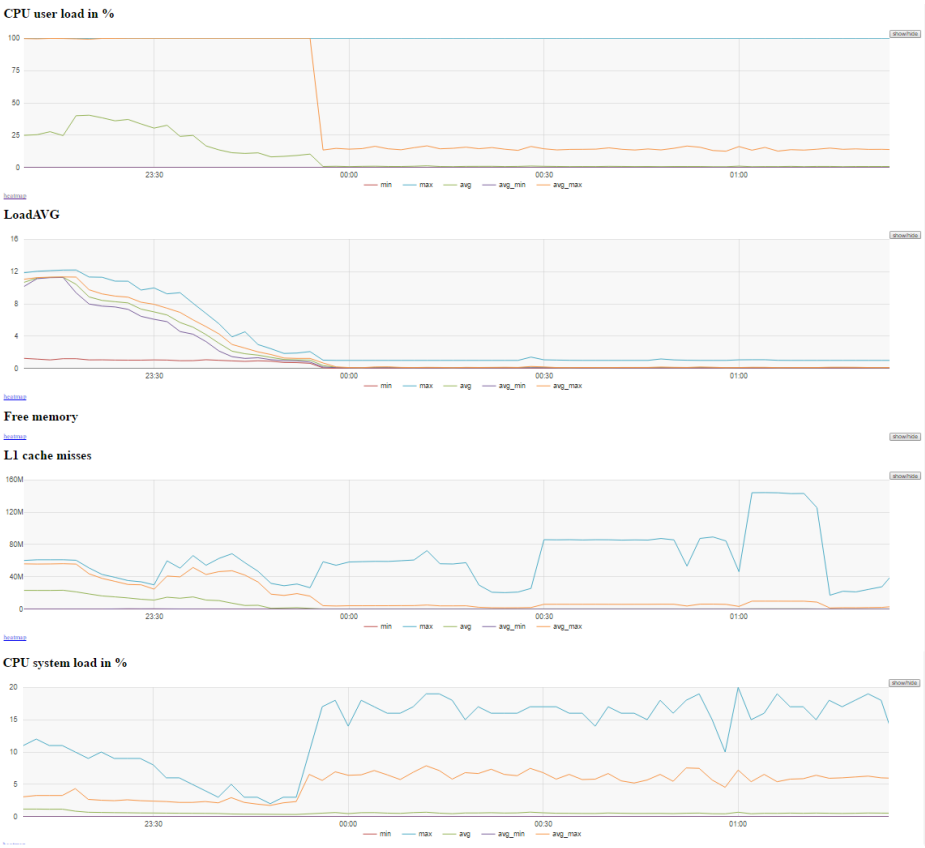

Fig. 7. JobDigest dynamic characteristic diagrams example.

The main goal of the JobDigest report is to detect bottlenecks and application performance degradation for any job in whole job flow, and special tools are required for further deep analysis to locate exact reasons in the program.

These diagrams are available in JobDigests as described in subsection B of section 3 of the paper.

Heat maps. Another useful way of analyzing dynamic characteristics in the JobDigest report is studying heat maps. Figure 8 represents the heat map for

CPU_user of the previous example. Only one node activity is clearly seen on the second stage of program run by the max values heat map, and looking at average values heat map it can be supposed that there was just one core used on that single node.

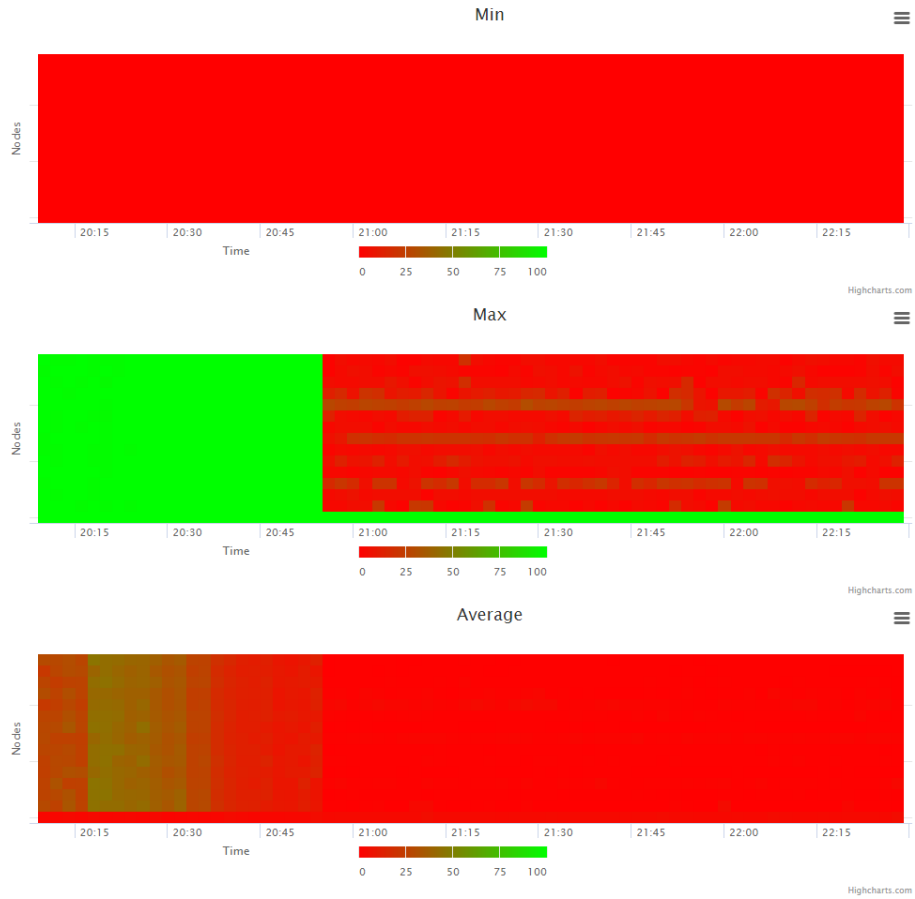

Fig. 8. JobDigest heat map for CPU_user dynamic characteristic.

Interesting examples of JobDigest usage are found everyday by support team of MSU supercomputer center. Some of those have been described in publications as a result of performed research on the reasons of performance degradation and specifics of resource utilization by various program types and program models [35].

7 Conclusions

The practiced approach based on system monitoring data analysis features possibility to have an in-depth view into every job launch without any changes to the program source codes or binaries to detect application performance issues.

The developed JobDigest reports provide valuable all-round view over the application execution behavior and resource utilization profile and rate.

All the described tools are developed in Moscow State University and are currently being used in the MSU HPC Center in 24/7 mode. The proposed system monitoring-based approach has been also evaluated in Uppsala University (Sweden) and showed promising results, Uppsala University team and SNIC support are hereby highly acknowledged.

The developed tools are available as an open source software, contributions are welcome.

Acknowledgements. The results were obtained in the Research Computing Center of M.V. Lomonosov Moscow State University. The work is funded in part by the Russian Found for Basic Research, grants №17-07-00719, №16-07-01121, and Russian Presidential study grant (SP-1981.2016.5).

References

1. Zenoss, http://www.zenoss.org, last accessed 2017/05/10.
2. Zabbix, http://www.zabbix.com, last accessed 2017/05/10.
3. Cacti®, http://www.cacti.net, last accessed 2017/05/10.
4. Massie, M. L. et al.: The ganglia distributed monitoring system: design, implementation, and experience. Parallel Computing, vol. 30(7), 817–840 (2004).
5. The OpenNMS project, http://www.opennms.org, last accessed 2017/05/10.
6. Nagios - the industry standard in IT infrastructure monitoring, http://www.nagios.org, last accessed 2017/05/10.
7. Collectd – The system statistics collection daemon, https://collectd.org, last accessed 2017/05/10.
8. Voevodin, Vl.V., Stefanov, K.S.: Distributed modular monitoring (DiMMon) approach to supercomputer monitoring. Proceedings of the 2015 IEEE International Conference on Cluster Computing, IEEE, 502–503 (2015).
9. Stefanov, K.S., Voevodin, Vl.V., Zhumatiy, S.A., Voevodin, Vad.V.: Dynamically Reconfigurable Distributed Modular Monitoring System for Supercomputers (DiMMon). Procedia Computer Science / Elsevier B.V., 2015, vol. 66, 625–634 (2015).
10. Gunter, D., Tierney, B., Jackson, K., Lee, J. and Stoufer M.: Dynamic monitoring of high-performance distributed applications. Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing, 163–170 (2002).
11. Mellor-Crummey, J., Fowler, R. J., Marin, G., and Tallent, N.: HPCVIEW: a tool for top-down analysis of node performance. The Journal of Supercomputing, vol. 23(1), 81–104 (2002).
12. Jagode, H., Dongarra, J., Alam, S. R., Vetter, J. S., Spear, W., and Malony, A. D.: A holistic approach for performance measurement and analysis for petascale applications. Computational Science – ICCS 2009, vol. 5545, 686–695 (2009).
13. Adhianto, L., Banerjee, S., Fagan, M., Krentel, M., Marin, G., Mellor-Crummey, J., and Tallent, N. R.: HPCTOOLKIT: tools for performance analysis of optimized parallel programs. Concurrency&Computation: Practice and Experience, vol. 22(6), 685–701 (2010).
14. Eisenhauer, G., Kraemer, E., Schwan, K., Stasko, J., Vetter, J. and Mallavarupu, N.: Falcon: on-line monitoring and steering of large-scale parallel programs. Proceedings of The Fifth Symposium on the Frontiers of Massively Parallel Computation, 422–429 (1995).

15. Kluge, M., Hackenberg, D. and Nagel, W. E.: Collecting distributed performance data with dataheap: generating and exploiting a holistic system view. Procedia Computer Science, vol. 9, 1969–1978 (2012).
16. Mooney, R., Schmidt, K. P., and Studham, R. S.: NWPerf: a system wide performance monitoring tool for large Linux clusters. 2004 IEEE International Conference on Cluster Computing (IEEE Cat. No.04EX935), 379–389 (2004).
17. Ries, B., et al.: The paragon performance monitoring environment. Supercomputing'93. Proceedings, 850–859 (1993).
18. Nikitenko, D.A.: Complex approach to performance analysis of supercomputer systems based on system monitoring data. Numerical methods and programming: New computing technologies, vol. 15, 85–97 (2014).
19. Antonov, A.S., Zhumatiy, S.A., Nikitenko, D.A., Stefanov, K.S., Teplov, A.M., Shvets, P.A.: Analysis of dynamic characteristics of job stream on supercomputer system. Numerical methods and programming: New computing technologies, vol. 14(2), 104–108 (2013).
20. Nikitenko, D.A., Adinets, A.V., Bryzgalov, P.A., Stefanov, K.S., Voevodin, Vad.V., Zhumatiy, S.A.: Job Digest - approach to analysis of application dynamic characteristics on supercomputer systems. Numerical Methods and Programming: New computing technologies, vol. 13, 160–166 (2012).
21. Voevodin, Vl.V., Voevodin, Vad.V.: Efficiency of exascale supercomputer centers and supercomputing education. High Performance Computer Applications: Proceedings of the 6th International Supercomputing Conference in Mexico (ISUM 2015). Springer. 14–23 (2016).
22. Nikitenko, D.A., Zhumatiy, S.A., Shvets, P.A.: Making Large-Scale Systems Observable — Another Inescapable Step Towards Exascale. Supercomputing Frontiers and Innovations, vol. 3(2), 72–79 (2016).
23. Shvets, P.A., Antonov, A.S., Nikitenko, D.A., Sobolev, S.I., Stefanov, K.S., Voevodin, Vad.V., Voevodin, Vl.V., Zhumatiy, S.A.: An approach for ensuring reliable functioning of a supercomputer based on a formal model. Parallel Processing and Applied Mathematics. 11th International Conference, PPAM 2015, Krakow, Poland, September 6-9, 2015. Revised Selected Papers, Part I, vol. 9573 of Lecture Notes in Computer Science, Springer International Publishing, 12–22(2016).
24. Nikitenko, D.A., Voevodin, Vl.V., Zhumatiy, S.A.: Resolving frontier problems of mastering large-scale supercomputer complexes. ACM International Conference on Computing Frontiers (CF'16), May 16-18, 2016, Como, Italy. ACM New York, NY, USA. 349–352 (2016).
25. Nikitenko, D.A., Voevodin, Vl.V., Zhumatiy, S.A.: Octoshell: Large Supercomputer Complex Administration System. Russian Supercomputing Days International Conference, Moscow, Russia, September 28-29, 2015. CEUR Workshop Proceedings, vol. 1482, 69–83 (2015).
26. Nikitenko, D.A., Stefanov, K.S., Zhumatiy, S.A., Teplov, A.M., Shvets, P.A., Voevodin, Vad.V.: System monitoring-based holistic resource utilization analysis for every user of a large HPC center, Algorithms and Architectures for Parallel Processing, LNCS, vol. 10049, Springer International Publishing. 305–318 (2016).
27. Slurm Workload Manager, http://slurm.schedmd.com, last accessed 2017/05/10.
28. Clustrx, http://www.hpcc.unical.it/hpc2010/ctrbs/tkachev.pdf, last accessed 2017/05/10.
29. JobDigest components. https://github.com/srcc-msu/job_statistics, last accessed 2017/05/10.

30. Mohr, B., Hagersten, E., Giménez, J., Knüpfer, A., Nikitenko, D., Nilsson, M., Servat, H., Shah, A., Voevodin, Vl., Winkler, F., Wolf, F., Zhukov, I.: The HOPSA workflow and tools. Proceedings of the 6th International Parallel Tools Workshop, Stuttgart, 2012
31. Voevodin, Vl.V., Zhumatiy, S.A., Sobolev, S.I., Antonov, A.S., Bryzgalov, P.A., Nikitenko, D.A., Stefanov, K.S., Voevodin, Vad.V.: Practice of "Lomonosov" Supercomputer. Open Systems J. - Moscow: Open Systems Publ., 7, 36–39 (2012).
32. Antonov, A.S., Teplov, A.M.: Generalized approach to scalability analysis of parallel applications. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10049 LNCS, 291–304 (2016).
33. Nikitenko, D.A., Voevodin, Vl.V., Voevodin, Vad.V., Zhumatiy, S.A., Stefanov, K.S., Teplov, A.M., Shvets, P.A.: Supercomputer application integral characteristics analysis for the whole queued job collection of large-scale HPC systems. 10th Annual International Scientific Conference on Parallel Computing Technologies, PCT 2016; Arkhangelsk; Russian Federation; March 29-31, 2016. vol. 1576 of CEUR Workshop Proceedings, 20–30 (2016).
34. Voevodin, Vl.V., Voevodin, Vad.V., Shaikhislamov, D.I., Nikitenko, D.A.: Data mining method for anomaly detection in the supercomputer task flow: Numerical Computations: Theory and Algorithms, The 2nd International Conference and Summer School, Pizzo calabro, Italy, June 20-24, 2016. AIP Conference Proceedings, vol. 1776, 090015-1-090015-4 (2016).
35. Andreev, D.Yu., Antonov, A.S., Voevodin, Vad.V., Zhumatiy, S.A., Nikitenko, D.A., Stefanov, K.S., and Shvets, P.A.: A system for the automated finding of inefficiencies and errors in parallel programs. Numerical methods and programming: New computing technologies, vol. 14(2), 48–53 (2013).