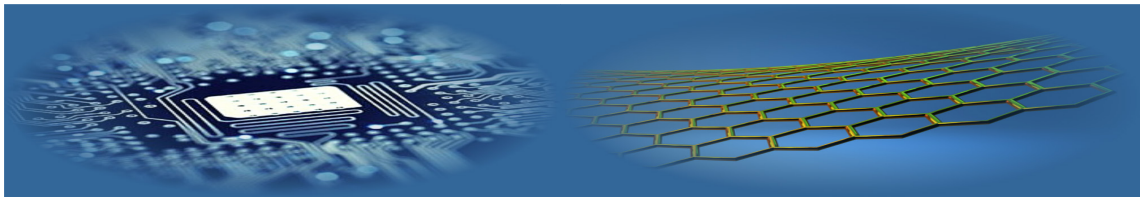

A study on MapReduce job failures in Hadoop

Ehsan Shirzad*, Hamid Saadatfar

Faculty of Electrical and Computer Engineering, University of Birjand, Daneshgah Blvd, Birjand, Iran

**Corresponding author's e-mail: e71.shirzad@gmail.com*

Received 30 March 2019, www.cmnt.lv

Abstract

Today, many big companies such as Facebook, Yahoo, and Google are using Hadoop for a variety of purposes. Hadoop is an open source software framework based on MapReduce parallel programming model for processing big data. Due to the importance of big data systems such as Hadoop, many studies have been conducted on these systems in order to achieve various goals such as efficient resource management, effective scheduling, and cognition of failure causes. By studying the failure causes, we can discern and resolve them, increase system's efficiency, and prevent from waste of resources and time. In this paper, we studied log files of a research cluster named OpenCloud in order to recognize job failures. OpenCloud has a long history of using Hadoop framework and has been used by researchers in various fields. Our study showed that different factors such as executing duration, number of executor hosts, volume of input/output data, and configurations affect the success or failure rate of the MapReduce jobs in Hadoop.

Key words

Log File; Hadoop; MapReduce; Cluster Workload; Job Failure

1 Introduction

Everything generates data, this concept is becoming more colorful since the advent of Internet. Nowadays, everything that connects to the Internet, from Internet users to electronic devices (IOT) is producing data. This huge generation of data caused scientists to encounter with enormous data that follows Moore's law (world's data is doubling every

two year) [1]. Thus, the amount of data stored around the world has reached 800000 petabytes in 2010 and it is expected to reach 35 zettabytes in 2020 [2] and 163 zettabytes in 2025 [3]. Some reasons for the rapid growth of data, which led to the emergence of big data concept, are new sources of information (such as mobile phones and sensors), increasing in the volume of generated information (such as video), and new subcategories of data (such as website clicks) [4]. Data in the 21st century is like oil in the 18th century, a source that needs to be refined [5].

Big data includes datasets that are generated from multiple sources at a high volume, velocity, and variety [6], which require new processing methods. Thus, in 2004, Google developed the MapReduce programming model to allow parallel and distributed processing of data [7]. In 2006, Yahoo and Apache developed an open source software framework based on MapReduce and called it Hadoop [8]. Hadoop is an Apache-based framework that combined with MapReduce parallel programming model to provide the possibility for users to handle large data sets on clusters consisting of multiple computers [9]. Therefore, many companies such as Amazon, Alibaba, Facebook, Google, LinkedIn, Twitter, and Yahoo are using Hadoop for various purposes (Powered by Apache Hadoop. https://wiki.apache.org/hadoop/PoweredBy).

In recent years, many studies have been conducted to improve the performance of big data systems and solve their problems. In summary, some of the works in this field are as follows: providing and sharing of log files (or trace data) to use in studies [10 - 13], predicting the availability of resources to improve resource management [14, 15], studying the behaviors of workloads to create balance in workload distributions [16 - 18] and to load prediction [19], studying the programming frameworks and the behavior of users in terms of the functions and configurations in order to improve the framework and add the required facilities for users [20, 21], anomaly detection [22], data skew detection [20, 21], job execution time and/or waiting time prediction for optimization of scheduling [23, 24], studying and analyzing the failures and its causes [25 - 32], and failure prediction [33 - 37].

As a definition of failure, a failure is an event that occurs when the delivered service deviates from accurate service (a misbehavior that can be observed by the user) [38]. Failures can cause serious problems for applications running on systems, such as performance degradation, data corruption, violation of service-level agreements, and ultimately losses of customers and revenue [12]. Furthermore, it is also evident that failures cause a waste of resources, energy and time. Therefore, a study on failure causes and factors can lead to an efficient system management in order to decrease the failures and ultimately saving the resources, increasing the system reliability, and increasing the user satisfaction. In this work, we tried to answer the following main questions:

- Does the job submission time affect the probability of failure?
- What impact do the execution-related characteristics of the jobs such as executing time and volume of input/output data have on job failures?
- What is the role of the hosts and communications between machines in success/failure of the jobs?
- What is the effect of the jobs configuration tunings on failures?

In order to study the MapReduce job failures in Hadoop as a big data system, we tried to analyse the first ten months of 2012 of the log files which belongs to OpenCloud cluster's workloads (CMU Cloud Computer Cluster at the Parallel Data Laboratory. Available: https://wiki.pdl.cmu.edu/OpenCloud). OpenCloud is a research cluster that belongs to Carnegie Mellon University. During the whole data collection period, the cluster included 64 nodes and each node had a 2.8 GHz dual quad core CPU (8 cores), 16 GB RAM, 10 Gbps Ethernet NIC, and four Seagate 7200 RPM SATA disk drives, and ran Hadoop 0.20.2. By using this log dataset, we studied some factors that affect job failures and found interesting patterns, which can be very useful for understanding the system behaviours and decreasing the failures.

The rest of this paper is organized as follows. Section 2 presents a survey of the previous works done in this area. Section 3 describes the log files. Section 4 presents the analyses of job failures in the studied system. Finally, Section 5 discusses the study results and concludes the paper.

2 Related works

One of the key points to achieve more efficiency in a large-scale processing system is

to decrease the failures. As mentioned above, failure is a misbehaviour that can be observed by the user and can cause losses of customers and revenue, and the waste of resources, energy, and time. One way to decrease the failures is to study the system log files in order to discover the causes of failures and eliminate them. A system's log files or workload trace data contain the events and their status which are occurred at a specific time in the system. In general, we can survey the works that have been done in the field of studying failures in large-scale distributed and parallel processing systems by using their workload trace data (or log files) in three aspects:

1. Study of the hardware characteristics (CPU/Memory); for example, Chen et al. [30] studied Google cluster's log files and computed the dependability of the cluster nodes by calculating the machine cycles (intervals that a machine was removed and added) to find the availability of the machines. They also visualized cluster resources usage by considering three factors of the jobs which are the priority class, multitasking, and life duration. Javadi et al. [12] also studied 9 trace datasets and investigated services unavailability intervals because of hardware failures. They calculated various statistical concepts (such as mean, median, standard deviation, and coefficient of variation) for the availability and unavailability intervals to find the central tendency, the spread, and the shape of their distribution. They discovered several distribution patterns for the unavailability intervals and modeled the hardware failures in the systems. Kondo et al. [39] also studied four real desktop grid trace files and modeled the resource availability for the task executions in the systems.
2. Study of the workload characteristics; for example, Kavulya et al. [25] studied the log files of Yahoo M45 Super Computing Cluster which used Hadoop. They investigated the workloads of the cluster in terms of date, number of jobs, number of users, research groups that used the cluster, mean and maximum of completion time of the jobs, and mean and maximum number of maps and reduces of the jobs. Chen et al. [21] studied the log files of 7 clusters of Facebook and Cloudera which used Hadoop. They investigated the workloads during one week through calculating number of the submitted jobs in an hour, total I/O size in the jobs, duration time of each job (total duration of map and reduce tasks), and the amount of cluster occupancy based on the number of active slots; and calculated the workloads bursting by using peak-to-median ratio of the cluster occupancy. Saadatfar et al. [32] analyzed the log files of a grid system called AuverGrid (part of the EGEE). They studied the impact of the workloads on job failures and discovered that the job failures increase by increasing in the number of waiting jobs in the queue, as well as increasing in the site load and reducing the number of free processors. Ren et al. [20] studied three cluster computing systems (including OpenCloud, M45 and Web Mining) that used Hadoop to process jobs. They investigated the applications in terms of the API and discovered that the users mostly used Hadoop native Java API and very rarely used higher level declarative interfaces such as Pig Latin. They also studied the workload types by using names of called functions. Rosa et al. [40] studied the Google cluster traces. They calculated the number of submitted, running and completed tasks in a time interval as the system workload; and studied the jobs status according to the priority class and the system load.
3. Study of the job characteristics; for example, Chen et al. [30] found that the distribution of the resources (CPU/Memory) consumptions of finished, failed and killed jobs in the cluster follows a heavy-tailed model. Kavulya et al. [25] investigated the completion time, the number of maps and reduces, and the status of the jobs in different months. They then calculated the Gini Coefficient to measure the equity of the tasks execution time. Saadatfar et al. [32] discovered that an increase in the memory usage increases the job failure. They defined a new hardware-related feature called CPU-intensity for the jobs, which an increase in this parameter indicates that the job spent

more lifetime on the processor and less time in the queue. They showed that the job failure increases by increasing in the CPU-intensity value. They also studied the failed jobs according to the month and day-hour of running time. Ren et al. [20] studied the user-defined configuration parameters of the jobs, so that, in each cluster, calculated the number of users that performed at least one additional function, in addition to map and reduce, as a statistic of job customization, and the number of users that performed at least one configuration tuning as a statistic of configuration frequency; therefore, they realized that Hadoop can add new facilities due to the needs of users. They also calculated the size of jobs' I/O data and their execution time as the amount of resource consumption. Rosa et al. [40] calculated the number of tasks, events, and running time for the jobs, as well as the queue waiting time, re-submission time, and executing time for the events. They also studied consumed resources (CPU/RAM/DISK) in the performed tasks in Google clusters and found some patterns in consumption of successful and unsuccessful tasks. Rosa et al. in another research on Google clusters [34], calculated the distribution patterns of successful and unsuccessful jobs with regard to the consumed resources. They then showed that the failed jobs have the highest consumption, which means that most resources of the system have been wasted.

Our study considers the three aspects. So that, we have tried to investigate characteristics related to jobs, workload, and resources and the correlation between them and job status.

In recent years, there have not been many works conducted to study Hadoop workloads' log files. Several works such as [20, 21, 25] studied Hadoop log files in order to investigate its performance, but they did not study the causes of failures. We studied some characteristics with respect to the similar studies, such as I/O volume data, submission time, and execution duration which can be found in [20, 25, 32, 40]; and we studied some novel characteristics according to our knowledge about the system, such as number of executor hosts and volume of communications. To the best of our knowledge, this is the first study that tried to investigate different aspects of failure causes in a real Hadoop cluster especially in OpenCloud.

3 Log dataset overview

The dataset has been collected from January 2012 until the end of October 2012, and each month separately has some tables in CSV format containing the features of jobs, tasks, and task attempts (hla. Available: ftp://ftp.pdl.cmu.edu/pub/datasets). In this study, we used five tables named conf, job, attempt, split, and counter, which are summarized in Tables (1) to (5). The jobs and task attempts have three statuses; success means successful completion, failed means failed completion (failure), and killed for the jobs, means stopping the jobs by users, and for the task attempts has three modes: if the user stops the task, if the executor node fails, or if they are speculative execution duplicates (when a task attempt takes too long, a new attempt runs, if the new one succeeds earlier, the previous attempt will be killed, and if the previous attempt succeeds, the new one will be killed). An attempt occurs when a task fails and it runs once more on another node (similar to task resubmission events in Google's MapReduce jobs [41]). Job Tracker is one of the main node elements and is responsible for distributing MapReduce tasks to particular nodes within a cluster. Shuffle is a step between map and reduce, which the maps' outputs are sorted first (sort step), then transferred to reduce step as input; in fact, the sort is a part of shuffle and the shuffle is a part of the reduce process. Split is referred to input splits of the map tasks. Hadoop also has many counters called built-in counters that each user can use them according to the needs [42].

TABLE 1 Overview of job configuration table (conf.csv)

Field Name	Description
jtid	Job tracker ID number
jobid	Job ID number
keyname	The key number assigned to each job configuration
value	The tuned value for the job configuration

TABLE 2 Overview of job history table (job.csv)

Field Name	Description
jtid	Job tracker ID number
jobid	Job ID number
submitTime	Job submission time (Unix timestamp format (starting from January 1, 1970. Available: www.unixtimestamp.com))
launchTime	Job start time (Unix timestamp format)
finishTime	Job finish time (Unix timestamp format)
status	Job completion status (0: successful, 1: failed, 2: killed)
numMaps	Total number of map tasks in the job
numReduces	Total number of reduce tasks in the job
finMaps	Total number of successful map tasks in the job
finReduces	Total number of successful reduce tasks in the job
failMaps	Total number of failed map tasks in the job
failReduces	Total number of failed reduce tasks in the job

TABLE 3 Overview of task attempt history table (attempt.csv)

Field Name	Description
jtid	Job tracker ID number
jobid	Job ID number
tasktype	Task type (m: map, r: reduce)
taskid	Task ID number
attempt	Task attempt number (start from 0 and increases with each additional attempt)
startTime	Attempt start time (Unix timestamp format)
shuffleTime	Shuffle start time (Unix timestamp format); only for reduce task
sortTime	Sort start time (Unix timestamp format); only for reduce task
finishTime	Attempt finish time (Unix timestamp format)
status	Attempt completion status (0: successful, 1: failed, 2: killed)
hostname	The ID number of the host that the attempt is running on it

TABLE 4 Overview of map split history table (split.csv)

Field Name	Description
jtid	Job tracker ID number
jobid	Job ID number
tasktype	Task type (map only)
taskid	Task ID number
splitid	Map's input split ID number (starts from 0)
split hostname	The ID number of the host that has the input data for the map task

TABLE 5 Overview of counter history table (counter.csv)

Field Name	Description
jtid	Job tracker ID number
jobid	Job ID number
tasktype	Task type (m: map, r: reduce); "?" for job-wide counters
taskid	Task ID number; "-1" for job-wide counters
attempt	Task attempt number; "-1" for task-wide counters
countergroup	Counter group number (for example, 4 for file-system counters)
counter	Counter ID number (for example, 6 for HDFS read bytes)
value	The value that obtained by the counter

4 Job failure analysis

An analysis and study of characteristics of jobs executed in a system can lead to recognizing the factors affecting jobs status and consequently the causes of failures. In the studied system, 21109 jobs were executed during ten months. Among the jobs, there were 18142 successful jobs (85.944%), 1980 failed jobs (9.3799%), and 981 killed jobs (4.647%). It should be noted that some of the jobs did not have a defined status or had a missed or wrong value in their table records; so, they were not included in the statistics that we presented in this section. The total execution time of the jobs (from submitting to end) is 25799051044 milliseconds (over 7166 hours), while, the total time of unsuccessful jobs (sum of the failed and killed job durations) is 10439469734 milliseconds (over 2899 hours). Therefore, according to the statistics, about 40.46 percent of the system time has been spent to execute the jobs was useless and it means wasting a lot of resources, energy, and time in the system. By studying the failed jobs, we can recognize the failure causes and eventually eliminate them; and as a result, we can save the resources, energy, and time.

Is this section, we investigate the jobs with regard to their different characteristics. We have studied the characteristics for all months' jobs because the cluster's workloads among the months were different. Figure 1 shows the number and status of the jobs during the first ten months of 2012. The most of the jobs were executed in February and the smallest amount of the jobs were executed in July and both were in an average level in terms of success and failure rates; 88.28 and 10.79%, respectively in February, and, 81.37 and 9.45%, respectively in July. The highest rate of failure was in September with 15.03%, and the highest rate of success was in January with 97.93%. The lowest rate of success was in August with 69.63%, meanwhile, the most killed jobs were in this month too with the rate of 21.11%.

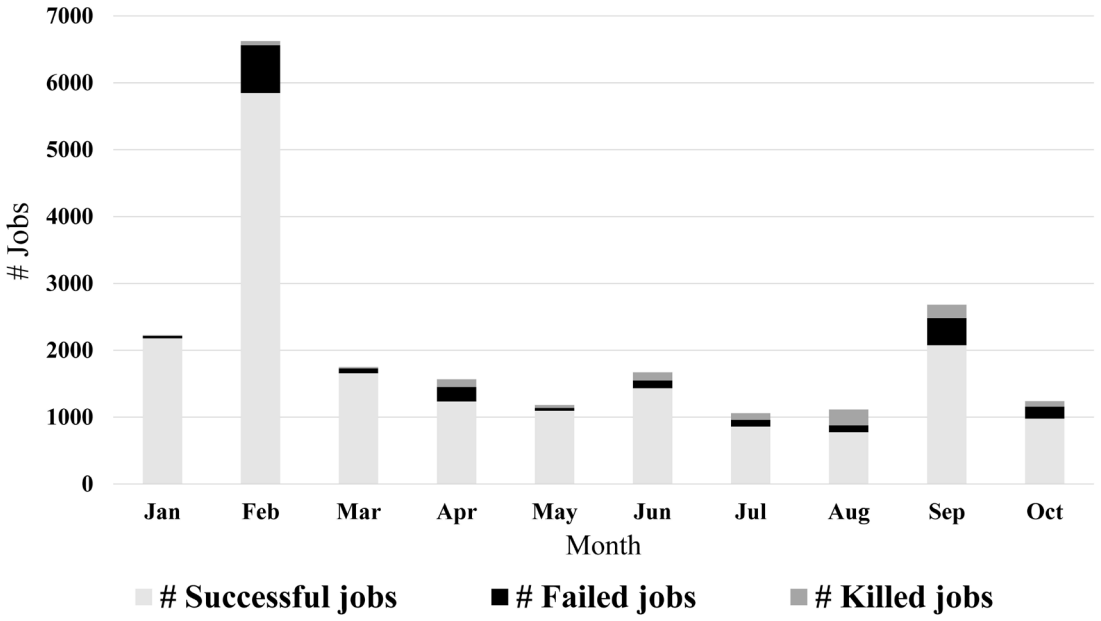

FIGURE 1 Distribution of the jobs during first ten months of 2012

Table 6 shows some basic statistics about the number of maps/reduces, the waiting time and the execution time of the jobs; which includes sum, mean, trimmed mean (the mean after discarding 5% of the maximum and minimum values), maximum, minimum, standard deviation, standard error of mean, and median. The statistics about the execution time shows that a small number of jobs had a very long execution time (42.23% of the jobs experienced under one minute of execution time, and only 4.02% of the jobs experienced more than one hour of execution time); this fact also can be seen for the waiting time (99.54% of the jobs had under one minute of waiting time). About the number of tasks, 14.48% of the jobs had only one map task, and 38.03% of the jobs had only one reduce task.

TABLE 6 Basic statistics about the jobs

Job's Property	Sum	Mean	Trimmed Mean	Min	Max	Standard Deviation	Std Error of Mean	Median
Number of Maps	11729411	580.57	291.41	0	375272	3395.42	23.88	63
Number of Reduces	631705	31.26	27.62	0	2470	49.14	0.34	10
Waiting Time (ms)	788323845	39020.13	15606.75	0	70000000	1001238.35	7044.16	15123
Execution Time (ms)	25010727199	1237970.95	183428.44	6365	494839893	11076066.29	77925.14	72762

The rest of this section includes studying the status of the jobs with regards to the submission time, execution time, input/output data volume, and executor host, and the impact of communications and configuration tuning on the job failure.

4.1 JOB STATUS AND SUBMIT TIME

In this section, we study the status of the jobs with regard to the days of the week and the hours of the day that the jobs were submitted to the system.

4.1.1 Job status and submit day hour

In order to know the number of submitted jobs at each hour of a day and study their condition, we converted the jobs' submit time in Table 2 into the day-hours according to the time of the state of Pennsylvania (Carnegie Mellon University location) and determined the jobs status at one-hour intervals. Figure 2 shows the jobs status at day-hours in percentage. The highest percentage of success was at 5-6 (92.02%), the lowest percentage of success was at 16-17 (79.58%), and the highest percentage of failure was at 7-8 (13.4%). Study of the workloads at these hours shows that the system has been working well in overcrowded conditions; because, the success rate was high at some hours with a large number of submitted jobs, such as 23-0, which had a success rate of 89.23%. Also, the number of submitted jobs was low at some hours with high rate of failure, such as 7-8. Unfortunately, there was no information about other related effective factors, such as details of the users and the working groups; we could say more about the submission time if this information was existed in the log files.

FIGURE 2 Job status for each hour of day

4.1.2 Job status and submit week day

Similar to previous section, we converted the jobs' submit time in Table 2 into the days of a week, and determined the jobs status on each week-day. Figure 3 shows the jobs status on each day of a week. The highest success rate and the lowest failure rate was on Sundays (weekend), and Mondays just had opposite condition. It is interesting to note that, totally, most of the submitted jobs was on Sundays (4072 jobs) and fewest number of the submitted jobs was on Monday (2037 jobs).

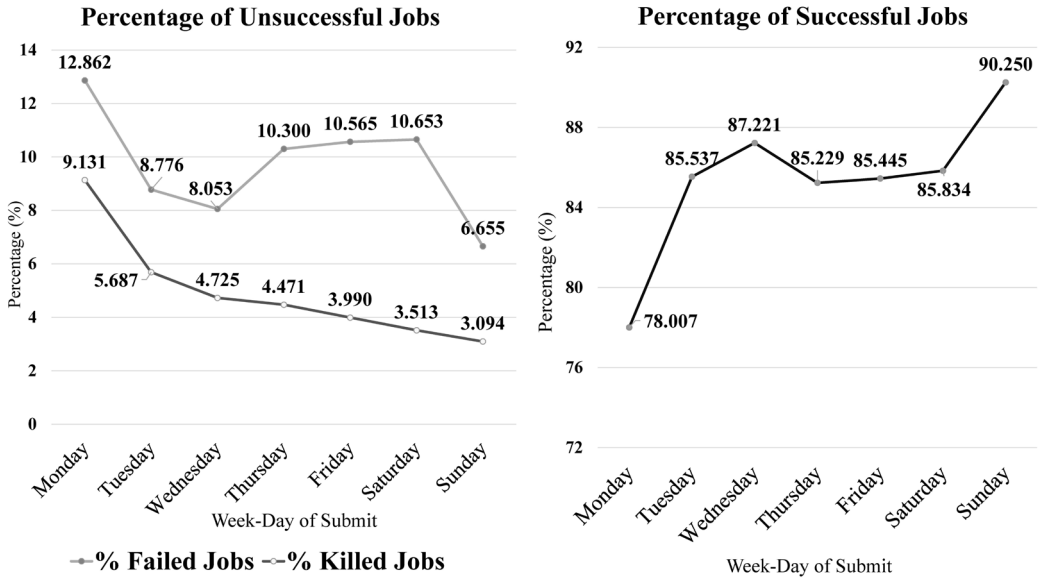

FIGURE 3 Job status for each day of week

4.2 JOB STATUS AND EXECUTION TIME

In this section, we calculated the jobs' executing duration by using start time and finish time fields in Table 2. Figure 4 shows the jobs status with regard to their execution time. The success percentage of the jobs decreases by increasing in the execution time, and the failure percentage has an increasing trend, except for one step. The percentage of the killed jobs by users severely increases for the jobs with the execution time of more than 10 minutes, which indicated that the users expected a quick response from the system. Figure 5 shows the cumulative distribution diagram of the status of the jobs in different execution times; it is observed that most of the jobs had less than 10 minutes of executing duration.

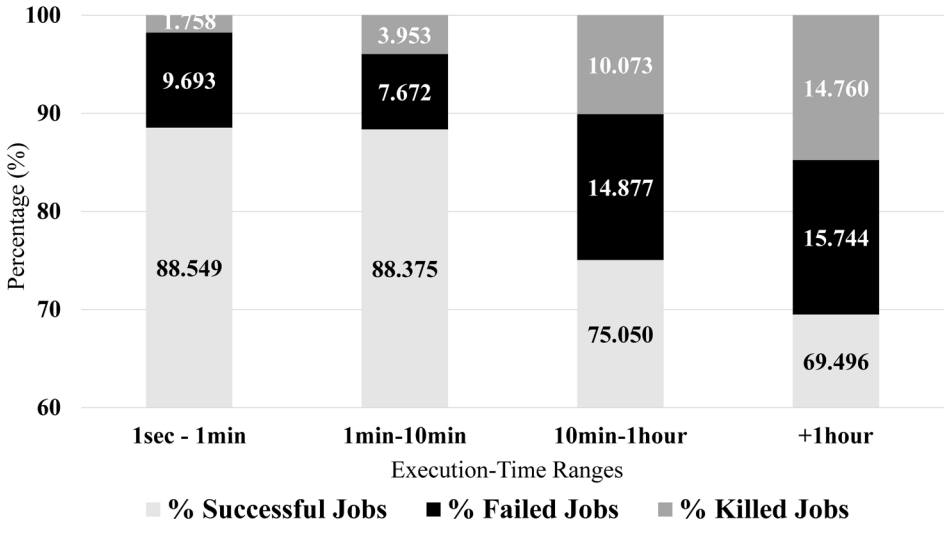

FIGURE 4 Job status for execution time intervals

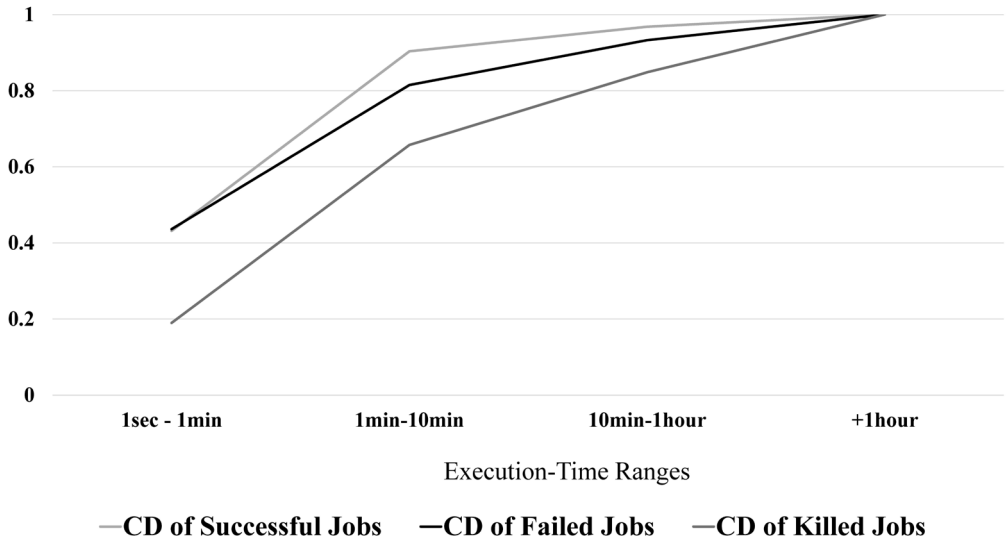

FIGURE 5 CD (Cumulative Distribution) of job status for execution time intervals

4.3 JOB STATUS AND I/O DATA VOLUME

In order to calculate the volume of input/output data and analyze it, we considered the counters of read and write bytes in the counters table (Table 5), and calculated the sum of the counters' values for each job as input/output data volume. Figure 6 shows the status of the jobs in I/O volume ranges. In Byte range (from one byte to less than one kilobyte) and KB range (from one kilobyte to less than one megabyte), the success rate of the jobs is almost the same and near 100%; but by increasing in I/O data volume, the success rate is reduced and the failure rate is generally increased, and in TB range (one terabyte of I/O volume and more), the success rate has been greatly decreased. The results indicate that working with memory (reading and writing data) is one of the factors affect the failure of the jobs in the system. Figure 7 shows that the I/O volume of most jobs was in the range of GB (from one gigabyte to less than one terabyte).

FIGURE 6 Job status for I/O data volume ranges

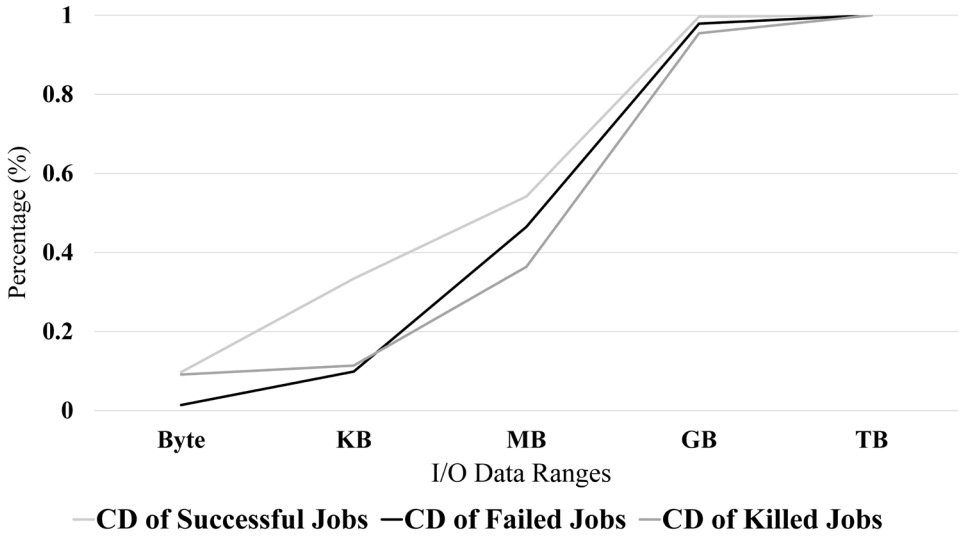

FIGURE 7 CD (Cumulative Distribution) of job status for I/O data volume ranges

4.4 JOB STATUS AND EXECUTOR HOSTS

When a MapReduce job is submitted to Hadoop, Hadoop divides it into different tasks (initially, all functions are maps and one or more reduces are performed after the completion of map process), and each task is given to a host to execute. If a task fails, the next executing attempt will be given to another host. In the log files of the studied system, the attempt history table (Table 3) contains the ID number of the hosts that execute the attempts (the hostname field). In order to study the impact of the number of executor hosts on failures, we joined job history table (Table 2) and attempt history table (Table 3), and calculated the number of distinct executor hosts for each job. Figure 8 shows the status of the jobs with regard to the number of executor hosts. It shows that by using more different hosts to execute a job's tasks, the probability of success increases; because, more available hosts to execute a job decreases the effect of a faulty host and consequently decreases the job failures.

Figure 9 shows the difference in the hosts according to the success/failure rate of the attempts executions (some hosts of Table 3 were randomly selected and analyzed). As can be seen in Fig. 9, the hosts have different success/failure rate in executing the tasks and it is because of the hardware specifications and software compatibilities. Unfortunately, the detailed information about the hosts was not existed in the log files, but we showed that the hosts have a significant impact on the success or failure of the tasks and consequently the jobs.

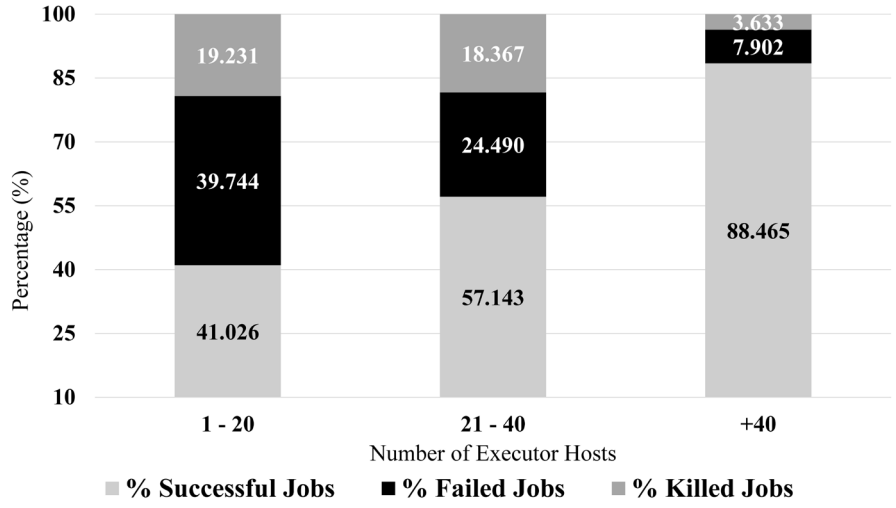

FIGURE 8 Job status with regard to the number of available executor hosts

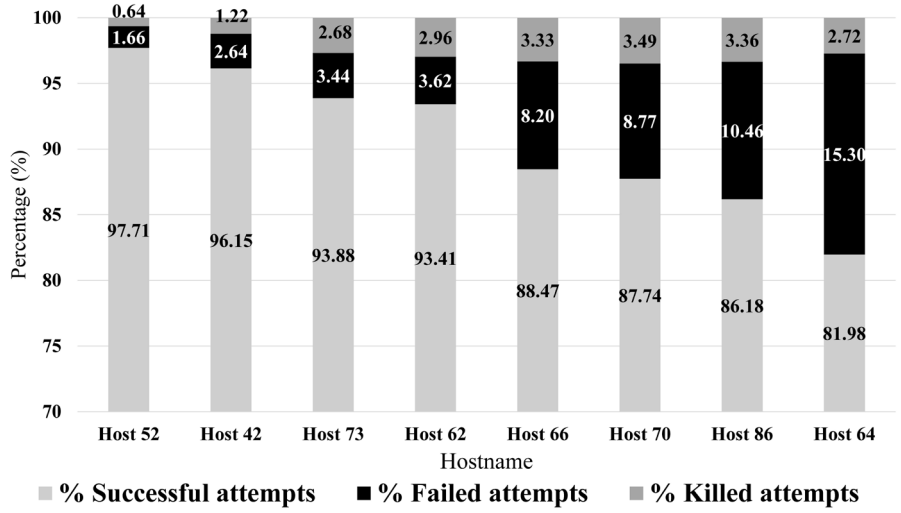

FIGURE 9 Differences in the success/failure rate of executor hosts

4.5 COMMUNICATION IMPACT ON FAILURES

In this section, we investigate the communication between the hosts that have the input data for map tasks and the executor hosts. As mentioned above, there are two types of logs for the hosts based on the tables: the log of the hosts which executed the attempts, and the log of the hosts which have the maps' input data splits. Therefore, when a map attempt is executed, the executor host must communicate with the host that stores the input data to receive the data (if the input data is not stored in the executor host) and perform the map process. We estimated the volume of communications by calculating the number of map input splits that was not stored in the executor host. Therefore, by using the attempt history (Table 3) and split history (Table 4) tables, for each map attempt, we calculated the number of the hosts that contained the input data but was distinct from the executor host (for example, one communication means that an attempt has one split which not stored in the executor host). Figure 10 shows the success percentage of the attempts with regard to the number of communications. As can be seen in Fig. 10, generally, an increase in the communications causes decreasing in success rate.

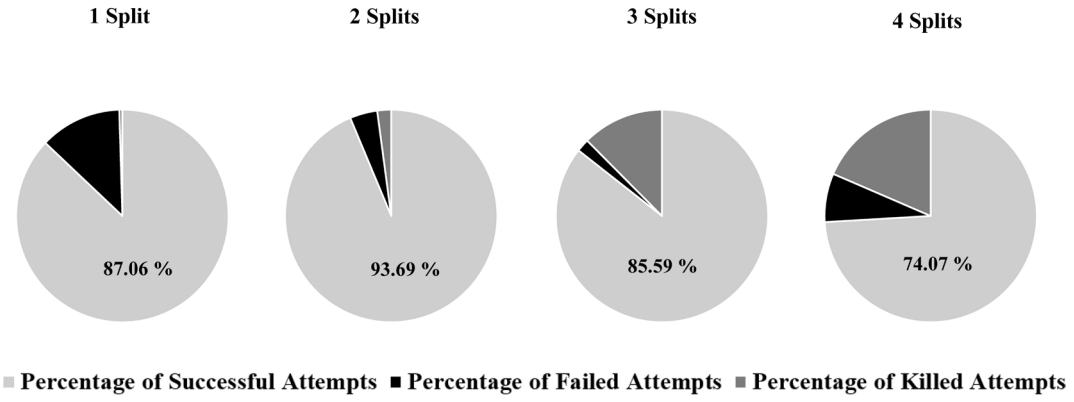

FIGURE 10 Attempt status with regard to the number of communications

4.6 IMPACT OF JOB CONFIGURATIONS TUNING ON FAILURE

One of the most important factors in the performance of a Hadoop system is the configuration parameters (Hadoop MapReduce Configurations. Available: https://hadoop.apache.org/docs/r1.2.1/mapred-default.html) which are tuned for the jobs, which if done correctly, can lead to optimal system performance and increasing in the success rate of the jobs. Therefore, many studies have been conducted on tuning the configurations

appropriately (such as [43, 44]). In this section, by using the configuration table (Table 1), we studied the configuration parameters which is tuned for the jobs. According to the table, many different configurations was tuned for the jobs in the system. According to the log files, most of the configurations are same for all jobs (probably was set by the admin), but we analyzed the rest of the job configurations and some of the results is explained in the rest of this section.

4.6.1. Number of attempts for jobs' tasks

A configuration parameter that has a significant impact on resource consumption is the number of determined task attempts (mapred.map.max.attempts to determine the number of map tasks and mapred.reduce.max. attempts to determine the number of reduce task), if this parameter is set to a high value without appropriate output, it causes multiple executions of a failed task and will cause the system resource wasting. According to the mentioned issue, in order to determine the optimal number of attempts for the relevant configuration, we analyzed the number of task attempts by using Table 3. Figure 11 shows the percentage of successful/failed attempts with regard to the number of executions; number 0 means the first execution of the task, number 1 means the first attempt to re-execute the task, and so on. As can be seen in Fig. 11, the success rate has been close to zero since attempt number 5, and all attempts are failed since attempt number 8. Therefore, increasing the number of execution attempts for the tasks is a useless operation since a juncture and it just wastes time and resources. Thus, there should be a limitation for tuning the number of task attempts in order to prevent the job's tasks from continuously useless executions.

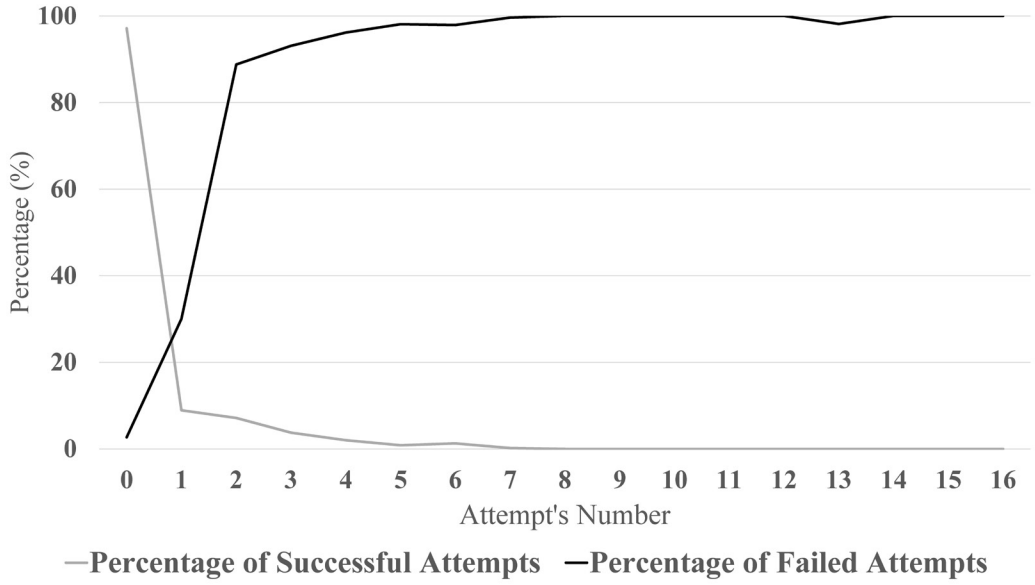

FIGURE 11 Attempt status with regard to the number

4.6.2 Size of virtual memory for jobs' tasks

According to the log files, another configuration that has an effect on the success of the jobs is the parameter for determining the amount of virtual memory in megabytes for the jobs' tasks (mapred.job.map.memory.mb for map tasks and mapred.job.reduce.memory.mb for reduce tasks). In order to study these parameters, we selected the values of the mentioned configurations in Table 1 for the jobs. Figure 12 shows the success/failure rate of the jobs with regard to the task memory size. As can be seen in Figure 12 the size of memory required for the tasks can be effective on jobs success or failure; and it is better to set the tasks' virtual memory size to 1024 MB or 4096 MB.

4.6.3 Number of acceptable skip records surrounding a bad record in map tasks

Another studied configuration is the number of acceptable skip records surrounding the bad record per bad record in mapper (mapred.skip. map.max.skip.record), which was checked for all jobs and found two values 0, and 256 for

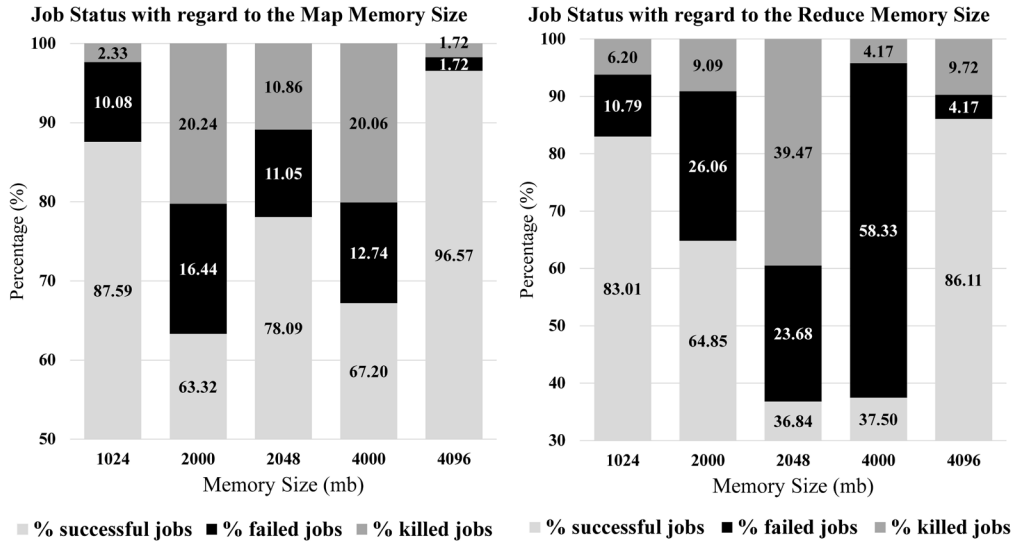

FIGURE 12 Job status with regard to required task memory size

them in the log files (a few jobs had no value (null) for this parameter which was removed from the study). We investigated the success/failure rate for the value 0 which means turning the feature of detection/skipping of bad records off, and the value 256 which means the threshold of 256 (Hadoop tries to narrow down the skipped range by retrying until this threshold is met or all attempts is done for the task). Figure 13 shows the jobs status with regard to the mentioned parameter. As can be seen, if this configuration is active and set to 256, it can be effective in decreasing job failures.

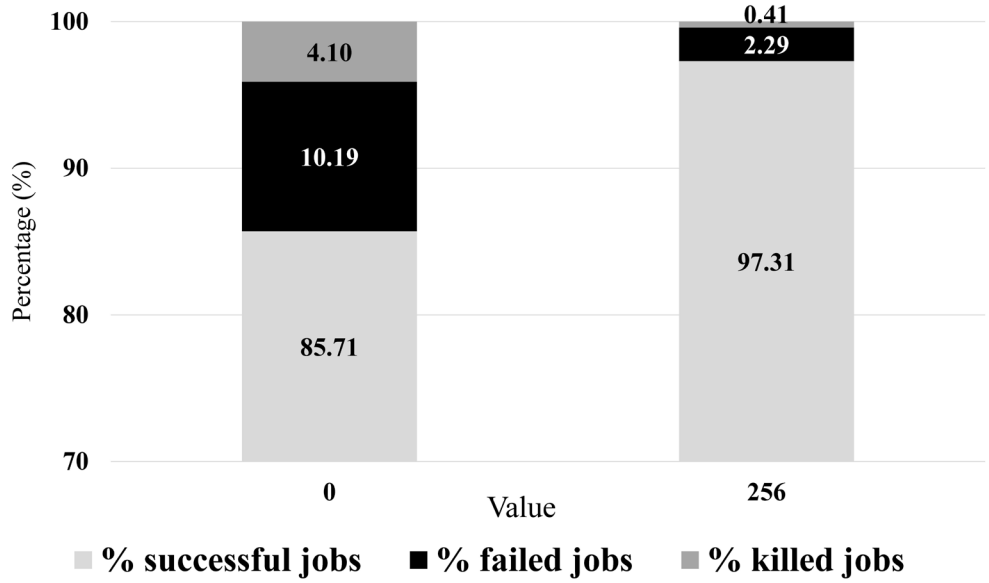

FIGURE 13 Job status with regard to the number of acceptable skip records surrounding a bad record in map tasks

5 Conclusions

In this paper, we studied the log files of OpenCloud, which is the research cluster at Carnegie Mellon University. This cluster is using Hadoop framework to process big data for many years. The dataset belongs to the first ten months of 2012, which was analyzed to discover the factors affecting success and failure of jobs. Studying the failures can help to recognize and eliminate the causes of them, increase efficiency and reliability, and decrease the waste of resources and time. As in the studied system, more than 40% of the system time has been spent for unsuccessful jobs.

We investigated various characteristics of

the jobs. It is showed that the highest success rate was in January and the lowest success rate was in August. In terms of submitting week-day, the highest failure rate was on Monday, and in terms of submitting day-hour, the highest failure rate was at 7-8. We also showed that an increase in the jobs execution duration increases the probability of their failure, and high volume of input/output data causes lower success rate of the jobs. It is also proved that if more different hosts are used to execute a job's tasks, the probability of success increases and an increase in the communications between the hosts decreases the success rate. Finally, we studied the impact of the jobs' configurations and showed that the value of 1024 MB or 4096 MB for the tasks' virtual memory sizes has a positive impact on the success rate of the jobs. Also, it is illustrated that the system needs a mechanism to restrict the number of re-execute attempts for the jobs' tasks, because, the probability of failure is close to 100% for the 5th attempt and higher. Furthermore, there was no success for the 8th attempt and upper which means re-executing a task for more than 8 times only wastes resources.

Although our study showed a lot of important characteristics of the jobs and system that have effects on the success or failure rate, it encountered some limitations including lack of hardware details (such as the memory and CPU characteristics of the hosts), the details of the racks, the machines' maintenance intervals, the jobs' scheduling priority, and the employed programming functions. If this information had been available, a more detailed analysis could have been performed.

Acknowledgments

The log files used in this study are obtained from the workloads of OpenCloud Hadoop cluster managed by CMU's Parallel Data Lab (hla. ftp://ftp.pdl.cmu.edu/pub/datasets).

References

[1] Bell G, Gray J, Szalay A 2006 Petascale computational systems *Computer* **39**(1) 110-2 doi: 10.1109/mc.2006.29
[2] Paul Zikopoulos CE 2011 *Understanding big data: Analytics for enterprise class hadoop and streaming data* 1st ed. New York: McGraw-Hill Osborne Media
[3] Huang S, Xu J, Liu R, et al. 2017 A novel compression algorithm decision method for spark shuffle process. *IEEE International Conference on Big Data*
[4] Cukier K 2010 Data, data everywhere: A special report on managing information *Economist Newspaper* Sect. 3-16 **E-source:** https://www.nytimes.com/2009/03/17/technology/business-computing/17cloud.html
[5] Van't Spijker A 2014 *The new oil: using innovative business models to turn data into profit* United States: Technics Publications
[6] Chen L, Gao S, Cao X 2017 Research on real-time outlier detection over big data streams *International Journal of Computers and Applications* 1-9 doi: 10.1080/1206212X.2017.1397388
[7] Dean J, Ghemawat S 2008 MapReduce: simplified data processing on large clusters *Communications of the ACM* **51**(1) 107-113. doi: 10.1145/1327452.1327492
[8] Vance A 2009 *Hadoop, a free software program, finds uses beyond search* New York Times. March
[9] Phu V N, Tran V T N 2018 K-Medoids algorithm used for english sentiment classification in a distributed system *Computer Modelling and New Technologies* **22**(1) 20-39
[10] Guo Y, Iosup A, et.al. 2012 The game trace archive *Proceedings of the 11th Annual Workshop on Network and Systems Support for Games* IEEE Press
[11] Iosup A, Li H, Jan M, et al. 2008 The Grid Workloads Archive *Future Generation Computer Systems* **24**(7) 672-86 doi: 10.1016/j.future.2008.02.003
[12] Javadi B, Kondo D, Iosup A, et al. 2013 The Failure Trace Archive: Enabling the comparison of failure measurements and models of distributed systems. *Journal of Parallel and Distributed Computing* **73**(8) 1208-23 doi: 10.1016/j.jpdc.2013.04.002
[13] Schroeder B, Gibson GA, et.al. 2007 The Computer Failure Data Repository (CFDR): collecting, sharing and analyzing failure data *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*
[14] Nadeem F, Prodan R, Fahringer T, et.al. 2008 Characterizing, Modeling and Predicting Dynamic Resource Availability in a Large Scale Multi-purpose Grid *8th IEEE International Symposium on Cluster Computing and the Grid*
[15] Rood B, Lewis M J 2008 Resource Availability Prediction for Improved Grid Scheduling 711-8 doi: 10.1109/eScience.2008.66
[16] Krompass S, Kuno H, Dayal U, et al. 2007 Dynamic workload management for very large data warehouses: Juggling feathers and bowling balls *Proceedings of the 33rd international conference on Very large data bases* VLDB Endowment
[17] Ke Q, Prabhakaran V, Xie Y, et al. 2011 Optimizing Data Partitioning for Data-Parallel Computing *HotOS* Napa, California
[18] Li H, Groep D, Wolters L, et.al. 2004 Workload characteristics of a multi-cluster supercomputer *Workshop on Job Scheduling Strategies for Parallel Processing* Springer
[19] Yuan Y, Wu Y, Yang G, et al. 2008 Adaptive hybrid model for long term load prediction in computational grid *8th IEEE International Symposium on Cluster Computing and the grid* IEEE
[20] Ren K, Kwon Y, Balazinska M, et al. 2013 Hadoop's adolescence: an analysis of Hadoop usage in scientific workloads *Proceedings of the VLDB Endowment* **6**(10) 853-64 doi: 10.14778/2536206.2536213
[21] Chen Y, Alspaugh S, Katz R 2012 Interactive analytical processing in big data systems: A cross-industry study of mapreduce workloads

Proceedings of the VLDB Endowment **5**(12) 1802-13 doi: 10.14778/2367502.2367519
[22] Gunter D, Tierney BL, Brown A, et al. 2007 Log summarization and anomaly detection for troubleshooting distributed systems *8th IEEE/ACM International Conference on Grid Computing* IEEE
[23] Kumar R, Vadhiyar S, et.al. 2014 Prediction of queue waiting times for metascheduling on parallel batch systems *Workshop on Job Scheduling Strategies for Parallel Processing* Springer
[24] Wu X, Zeng Y, Zhao C 2015 Regression-based execution time prediction in Hadoop environment Information, Computer and Application Engineering *Proceedings of the International Conference on Information Technology and Computer Application Engineering. Hong Kong, China: Informa UK Limited* 623-7
[25] Kavulya S, Tan J, Gandhi R, et al. 2010 An analysis of traces from a production mapreduce cluster *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing* IEEE Computer Society
[26] Samak T, Gunter D, Goode M, et al. 2012 Failure analysis of distributed scientific workflows executing in the cloud *Proceedings of the 8th international conference on network and service management* Las Vegas, Nevada: International Federation for Information Processing
[27] El-Sayed N, Schroeder B, et.al. 2013 Reading between the lines of failure logs: Understanding how HPC systems fail *43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks* IEEE
[28] Schroeder B, Gibson G 2010 A large-scale study of failures in high-performance computing systems *IEEE Transactions on Dependable and Secure Computing* **7**(4) 337-50 doi: 10.1109/tdsc.2009.4
[29] Rosa A, Chen L Y, Binder W, et.al. 2015 Understanding the dark side of big data clusters: An analysis beyond failures *45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks* IEEE
[30] Chen X, Lu C-D, Pattabiraman K, et.al. 2014 Failure analysis of jobs in compute clouds: A google cluster case study *IEEE 25th International Symposium on Software Reliability Engineering* IEEE
[31] Klusáček D, Parák B, et.al. 2017 Analysis of Mixed Workloads from Shared Cloud Infrastructure *Workshop on Job Scheduling Strategies for Parallel Processing* Springer
[32] Saadatfar H, Fadishei H, Deldari H 2012 Predicting job failures in AuverGrid based on workload log analysis *New Generation Computing* **30**(1) 73-94 doi: 10.1007/s00354-012-0105-z
[33] Fadishei H, Saadatfar H, Deldari H, et.al. 2009 Job failure prediction in grid environment based on workload characteristics *14th International CSI Computer Conference* IEEE
[34] Rosa A, Chen LY, Binder W, et.al. Predicting and mitigating jobs failures in big data clusters *15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing* IEEE
[35] Liu C, Han J, Shang Y, et al. 2017 Predicting of Job Failure in Compute Cloud Based on Online Extreme Learning Machine: A Comparative Study *IEEE Access* **5** 9359-9368 doi: 10.1109/access.2017.2706740
[36] Chen X, Lu C-D, Pattabiraman K, et.al. 2014 Failure prediction of jobs in compute clouds: A google cluster case study *IEEE International Symposium on Software Reliability Engineering Workshops* IEEE
[37] Hongyan T, Ying L, Long W, et al. 2017 Predicting Misconfiguration-Induced Unsuccessful Executions of Jobs in Big Data System *IEEE 41st Annual Computer Software and Applications Conference* IEEE
[38] Salfner F, Lenk M, Malek M 2010 A survey of online failure prediction methods *ACM Computing Surveys* **42**(3) 10 doi: 10.1145/1670679.1670680
[39] Kondo D, Fedak G, Cappello F, et al. 2007 Characterizing resource availability in enterprise desktop grids *Future Generation Computer Systems* **23**(7) 888-903 doi: 10.1016/j.future.2006.11.001
[40] Rosa A, Chen LY, Binder W 2017 Failure analysis and prediction for big-data systems *IEEE Transactions on Services Computing* **10**(6) 984-98 doi: 10.1109/TSC.2016.2543718
[41] Reiss C, Wilkes J, Hellerstein JL 2011 *Google cluster-usage traces: format+ schema* Google Inc, White Paper 1-14
[42] White T 2012 *Hadoop: The definitive guide* Yahoo Press, USA: " O'Reilly Media, Inc."
[43] Lama P, Zhou X, et.al. 2012 Aroma: Automated resource allocation and configuration of mapreduce environment in the cloud *Proceedings of the 9th international conference on Autonomic computing* ACM
[44] Babu S, et.al. 2010 Towards automatic optimization of MapReduce programs *Proceedings of the 1st ACM symposium on Cloud computing* ACM

AUTHORS

Ehsan Shirzad, 6 August 1992, Mashhad, Iran

Current position, grades: MSc in Information Technology, department of computer engineering
University studies: University of Birjand
Scientific interest: Data mining, Machine learning, Big data, Data analysis, Internet of things
Publications: 1 article in national conference and 1 article in national journal
Experience: 2 years of researching and 1 year of teaching

Hamid Saadatfar, 30 November 1984, Mashhad, Iran

Current position, grades: grades: assistant professor of computer engineering department, Ph.D. in Computer engineering
University studies: University of Birjand
Scientific interest: Distributed and parallel computing, Machine learning, Big data
Publications: More than 15 articles in national and international journals and conferences
Experience: 12 years of researching and 8 years of teaching