Investigation and Characterization of MapReduce Applications for Big Data Analytics

Y Li¹², T B V Lam¹, T V Van Do³⁴*, R Chakka⁵ and C Rotter⁶

¹Department of Networked Systems and Services, Budapest University of Technology and Economics, H-1117, Magyar tudósok körútja 2, Budapest, Hungary
²Department of Computer Science, Xi’an Siyuan University, 710038, Xi’an, China
³Division of Knowledge and System Engineering for ICT (KSE-ICT), Ton Duc Thang University, HCM City, Vietnam
⁴Analysis, Design and Development of ICT systems (Ad stilICT) Laboratory, Budapest University of Technology and Economics, H-1117, Magyar tudósok körútja 2, Budapest, Hungary
⁵Rajeev Gandhi Memorial College of Engineering and Technology (RGMCET), Nandyal-518501, India
⁶Nokia Bell Nokia Bell Labs Hungary, Bókay János 36-42, Budapest, Hungary

Received 24 September 2017; revised 14 March 2018; accepted 17 June 2018

Recently, many organisations have applied Hadoop MapReduce framework for big data analytics. MapReduce applications based on the MapReduce programming model can be developed to process data of large amount. Therefore, understanding a dependency among the resource usage parameters of MapReduce applications is crucially needed from the viewpoint of cloud operators. In this paper, we analyze the inter-dependency of resource usage parameters of MapReduce applications. Autocorrelation of each resource usage parameter and correlation characteristics of each pair of resource usage parameters are investigated. Based on the analysis, we identify several groups of features that can be used to classify MapReduce applications.

Keywords: Resource Usage Parameter, Mapreduce Application, Autocorrelation, Correlated Characteristic, Read-Intensive, Write-Intensive, CPU-Intensive, Read/Write Intensive

Introduction

Big data analytics can lead to useful results in many areas and fields.¹² MapReduce is a programming model of Apache Hadoop³, which allows developers to write their MapReduce applications for big data analytic purposes. A specific MapReduce application may be CPU-intensive, read-intensive, write-intensive, or read/write-intensive. Therefore, understanding a dependency among the resource usage parameters of each of the MapReduce applications is crucially needed.⁴⁵⁶⁷ In this paper, we aim the investigation of various characteristics that can be used to classify MapReduce applications such as Word count, Wordmean, Word median, Grep, Pi, Teragen, and Terasort. We analyse the correlation and the autocorrelation characteristics of resource usage parameters, such as CPU usage, memory usage, read rate and write rate. We identify some key features that can be used to identify MapReduce applications. To the best of our knowledge, the paper is the first attempt to reveal the dependence of the usage parameters of various MapReduce applications.

Data Collection

The workload for those read-intensive applications is generated with the use of hdfswriter.jar (written in Java). The workload is a text file of 100GB for read-intensive applications. The workload of Terasort is 60GB data generated from Teragen. We capture the usage parameters of Teragen when it generates 120GB data. We run Pi with 2000 map tasks in 10000000 times. Collect utility⁸ measures the total percentage of time spent of CPU processing job, the total memory usage, the total KB read/second from hard disk and the total KB write/second to the hard disk of MapReduce applications with time resolution 1 s.

Explore Non-randomness

The autocorrelation function⁹ and the auto covariance function are applied to explore the non-randomness of each resource usage parameter. Let \( x_t \) denote the value of a time series at time instant \( t \).
The autocorrelation between \( x_i \) and \( x_{i+k} \) is given by autocorrelation coefficient, denoted by \( \rho_k \), for \( k=\{1,2,3,\ldots,n\} \).

\[
\rho_k = \frac{\sum_{i=1}^{n-k}(x_i-x)(x_{i+k}-\bar{x})}{\sum_{i=1}^{n}(x_i-x)^2} \ldots (1)
\]

For a stochastic process \( X=\{X_i\} \), the auto covariance is a function that gives the covariance\(^{10} \) of the process with itself at different time points. The auto covariance plot shows an absolute linear association degree between points in time series data. The Ljung–Box Q test\(^{11} \) is used to identify the stationary of time series data. Note that non-randomness is significant if there is at least one autocorrelation violates the 95% confidence interval.

**Correlated characteristic analysis**

Pearson correlation coefficient\(^{12} \) is a commonly-used metric to measure a linear correlation between a pair of variables. Pearson correlation coefficient is denoted by

\[
r = \frac{\sum_{i=1}^{n-k}(x_i-x)(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-x)^2} \sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}} \ldots (2)
\]

Where \( n \) is the number of samples, \( x_i \) and \( y_i \) are the single samples indexed with \( i \), while \( \bar{x} \) and \( \bar{y} \) are the averages of samples.

**Experimental environment**

The basic experimental setting is as follows.

- Bare metal server with an Intel Core TM i5-4670 CPU 3.40GHz 4 cores, 16GB Kingston HyperX Black DDR3 1600MHz RAM and 250GB 7200RPM hard drive is used.
- Ubuntu server 16.04.3 LTS, kernel 4.4.0-62-generic, has been used to the top of the physical hardware.
- The Hadoop single node mode is installed. The node runs default configuration of Hadoop version 2.7.3 and MapReduce v2. The block size is set to be 512 MB.

**Non-randomness Identification**

In Figure 1 we plot the autocorrelation and the auto covariance of the resource usage parameters of Pi application (CPU-intensive). The top half panel of Figure 1 shows that both the CPU usage and the memory usage reveal non-randomness because of autocorrelations, denoted by the circle, violating the dashed lines and being statistically significant for lags up to 100. The filled triangle point-up marks the autocorrelation of the CPU usage and its lag 1 values at 0.983. The autocorrelation for the memory usage is 0.932. These values show highly positive relevance between the CPU usage and its lag 1 values as well as between the memory usage and its lag 1 values. Meanwhile, the auto covariance of the CPU usage and that of the memory usage with their lag 1 values are 1516.341 and 16.877 respectively, which shows the significant non-zero quantity. On the bottom left panel of Figure 1, although there is high autocorrelation (0.732) between the read rate and its lag 1 values, the corresponding auto covariance is 0.029. This result indicates that the read rate is non-random. The same observation can be obtained with the write rate. The Terasort application exhibits the significant non-randomness on the CPU usage, the memory usage, the read rate and the write rate. When compared with Pi application, the auto covariance plots of the read rate and the write rate reveal the salient difference. The auto covariance of Terasort between the read rate and its lag 1 values shows the strong dependency. The write rate has the similar characteristics with that of the read rate. The CPU usage, the memory usage of CPU-intensive applications such as Pi exhibit significant non-randomness. The strong linear dependency exists between these parameters and their lag 1 values. The CPU usage, the memory usage and the read rate of the read-intensive applications, such as Word count, Wordmean, Word median and Grep, exhibit strong non-randomness and linear dependency. The CPU usage, the memory usage and the write rate of write-intensive applications like Teragen is non-random and linear dependent, while the read rate exhibits non-randomness and dependency. The resource usage parameters of the read/write-intensive applications, such as Terasort, reveal the extremely significant non-randomness and strong linear dependency. Note that the strongest autocorrelations happen between these resource usage parameters and their lag 1 values.

**Correlation matrix**

Table 1 shows the correlation matrix between the usage parameters of Pi application (CPU-intensive), Word count (read-intensive), Teragen (write-
DO et al: INVESTIGATION AND CHARACTERIZATION OF MAPREDUCE APPLICATIONS FOR BIG DATA

intensive) and Terasort (read/write-intensive). The correlation between the CPU usage and the memory usage exhibits the extremely strong positive linear relevance (0.957). There is not any significant relationship between write rate and others. The relationship between the read rate and other usage parameters is very weak, and the correlation coefficients are always negative. The autocorrelations of the CPU usage (0.627), the memory usage (1) and the read rate (0.61) of Word count shows the strong positive linear relevance while the write rate (-0.018) exhibits almost no relationship. The correlation between the CPU usage and the memory usage shows weak positive relevance (0.142) and between the read rate and the write rate is of weak negative relevance (-0.169). The correlation coefficients of the resource usages of Teragen, which is a write-intensive MapReduce application, are listed in Table 1 as well.

Due to the read-intensive characteristic, the correlation coefficients between CPU usage (0.228) or memory usage (0.279) and read rate show weak positive relevance, while the write rate presents almost no relationship. The correlation between the CPU usage and the memory usage is -0.396, which shows a moderate negative relevance. Furthermore, there are almost no relationships between the CPU usage and the read rate (0.077) or the write rate (0.007). In contrast, the correlations between the memory usage and the read
rate (-0.436), and the memory usage and the write rate (0.238) exhibit a moderate negative relevance and a weak positive relevance. The Read rate and the write rate has a moderate negative association (-0.415). Terasort application is read-intensive as well as write-intensive. All of the autocorrelations coefficients between resource usage parameters of Terasort and their lag 1 values exhibit a highly positive relevance, and the smallest coefficient is 0.692. The correlation is significant between the read rate and the write rate, which shows moderate negative relevance (-0.495). The correlation between the memory usage and the read rate shows a weak negative relevance (-0.269). The remaining correlations all exhibit extremely weak relationships or even no relationship.

**Analysis and Discussion**

Our results show that MapReduce applications with different resource intensive characteristics do not exhibit the similar characteristics on correlation and autocorrelation. The exception is a high value associated with the auto correlation coefficient of the memory usage because MapReduce applications often reserve a memory to store and process data. There is no obvious relationship between the write rate and other resource usage parameters in the CPU-intensive applications and the read-intensive applications, while other two classes of applications present a strong correlation. One possible explanation is that the write rate plays an insignificant role in these applications, e.g. Pi (CPU-intensive) and Wordcount (read-intensive). The correlation coefficient between the CPU usage and the memory usage of CPU-intensive applications is very high, while such metric in read/write-intensive ones is very low. If an application belongs to a certain type of intensive resource, this resource will have a high value of autocorrelation coefficient and have a certain relationship with the CPU usage and the memory usage. Based on some common signatures about correlation coefficient and autocorrelation of resource usage parameters, we can identify the resource-intensive categorisation to which the MapReduce application belongs. That is, thresholds for the correlation coefficient are defined in Table2. According to the different resource-intensive types, the distributions of these correlation coefficients and autocorrelation are plotted in Figure2. The different types of dashed lines correspond the threshold levels (see Table2). For all tested MapReduce applications, they perform the same characteristics on the perfect autocorrelation of memory usage and strong positive autocorrelation of the read rate. Except for these two common characteristics, MapReduce applications with various resource-intensive classes present many different signatures.

On the left panel of Figure2, the correlation coefficient between the CPU usage and the memory usage as well as autocorrelation of the CPU usage shows an extremely high value, which is larger than 0.9. Meanwhile, the read rate and the write rate show almost no relationship because the absolute value of correlation coefficient is less than 0.1.

For CPU-intensive MapReduce applications, it can be observed that

- the autocorrelation coefficient of the CPU usage is positive high,
- the correlation coefficient of the CPU usage and memory usage is positive high.

<table>
<thead>
<tr>
<th>Application Name</th>
<th>Usage parameter</th>
<th>CPU usage</th>
<th>memory usage</th>
<th>read rate</th>
<th>write rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pi</td>
<td>memory usage</td>
<td>0.957</td>
<td>-0.122</td>
<td>-0.156</td>
<td>0.047</td>
</tr>
<tr>
<td></td>
<td>read rate</td>
<td>-0.122</td>
<td>0.079</td>
<td>0.067</td>
<td>-0.02</td>
</tr>
<tr>
<td></td>
<td>write rate</td>
<td>0.079</td>
<td>0.984</td>
<td>0.933</td>
<td>0.732</td>
</tr>
<tr>
<td></td>
<td>lag 1 autocorrelation</td>
<td>0.984</td>
<td>0.933</td>
<td>0.732</td>
<td>-0.025</td>
</tr>
<tr>
<td>Wordcount</td>
<td>memory usage</td>
<td>0.142</td>
<td>0.228</td>
<td>0.279</td>
<td>-0.002</td>
</tr>
<tr>
<td></td>
<td>read rate</td>
<td>0.228</td>
<td>0.002</td>
<td>0.005</td>
<td>-0.169</td>
</tr>
<tr>
<td></td>
<td>write rate</td>
<td>-0.002</td>
<td>0.627</td>
<td>1</td>
<td>-0.61</td>
</tr>
<tr>
<td></td>
<td>lag 1 autocorrelation</td>
<td>0.627</td>
<td>1</td>
<td>-0.169</td>
<td>-0.018</td>
</tr>
<tr>
<td>Teragen</td>
<td>memory usage</td>
<td>-0.396</td>
<td>-0.077</td>
<td>-0.436</td>
<td></td>
</tr>
<tr>
<td></td>
<td>read rate</td>
<td>-0.077</td>
<td>0.007</td>
<td>0.238</td>
<td>-0.415</td>
</tr>
<tr>
<td></td>
<td>write rate</td>
<td>0.007</td>
<td>0.707</td>
<td>1</td>
<td>0.822</td>
</tr>
<tr>
<td></td>
<td>lag 1 autocorrelation</td>
<td>0.707</td>
<td>1</td>
<td>0.822</td>
<td>0.781</td>
</tr>
<tr>
<td>Terasort</td>
<td>memory usage</td>
<td>-0.089</td>
<td>0.046</td>
<td>-0.269</td>
<td></td>
</tr>
<tr>
<td></td>
<td>read rate</td>
<td>0.046</td>
<td>0.048</td>
<td>-0.073</td>
<td>-0.495</td>
</tr>
<tr>
<td></td>
<td>write rate</td>
<td>0.048</td>
<td>0.692</td>
<td>0.995</td>
<td>0.893</td>
</tr>
<tr>
<td></td>
<td>lag 1 autocorrelation</td>
<td>0.692</td>
<td>0.995</td>
<td>0.893</td>
<td>0.834</td>
</tr>
</tbody>
</table>
One can observe from Figure 2 that read-intensive applications have similar correlation characteristics on three pairs of variables (the memory usage and the read rate), (the read rate and the write rate), (the memory usage and the write rate) and the autocorrelation coefficient of the write rate.

For Read-intensive MapReduce application, one can observe the following signatures:
- the autocorrelation coefficient of the read rate is positive high,
- the autocorrelation coefficient of the write rate is very low (In other words, we can say there is randomness in the values of write rate),
- the correlation coefficient of the read rate and the memory usage is positive and at least moderate,
- the correlation coefficient of the write rate and the memory usage is very low.

On the left panel of Figure 2, the autocorrelations of the write rate and the read rate show the strong positive relevance. The correlation between the read rate and the write rate is significantly negative.

For write-intensive MapReduce application, the observations are as follows:
• the autocorrelation coefficient of the write rate is positive high,
• the correlation coefficient of the write rate and the memory usage is positive,
• the correlation coefficient of the read rate and the memory usage is negative.

For read/write-intensive MapReduce application, we observe the following signatures:
• the autocorrelation coefficient of the write rate is positive high,
• the correlation coefficient of the read rate and either the CPU usage or the memory usage is positive,
• the correlation coefficient of CPU usage and memory usage is low.

It is observed that not all MapReduce applications have the similar correlation characteristics on resource usage parameters. Additionally, the results reveal the signatures of different resource-intensive classes of MapReduce applications. In summary, the correlation and autocorrelation of resource usage parameters are quite good to characterise MapReduce applications. It is worth emphasising that we have made the same analysis when the dynamic CPU frequency scaling mechanism is applied, and we got the same observations as well.

Conclusion
We have exposed the relationship of resource usage parameters, namely, the CPU usage, the memory usage, the read rate and the write rate as well as autocorrelation pattern of each resource usage parameter. We have shown that MapReduce applications might be categorised based on the correlation coefficients of resource usage parameters. Some potential research directions are arising from our study. For example, we can take into account the relationship of the resource usage parameters of MapReduce applications to establish prediction models for the resource usages.

References
6. Nghiem P P & Figureueira S M, Towards efficient resource provisioning in MapReduce, J Pur Dist Com, 95(C) (2016) 29-41,