To Default or not to Default: Exposing limitations to HBase Cluster Deployers

Conference Paper · November 2015

CITATIONS
0

READS
16

5 authors, including:

Marios Fokaefs
York University
31 PUBLICATIONS **287** CITATIONS
SEE PROFILE

Hamzeh Khazaei
University of Alberta
50 PUBLICATIONS **634** CITATIONS
SEE PROFILE

Marin Litoiu
York University
173 PUBLICATIONS **3,484** CITATIONS
SEE PROFILE

To Default or not to Default: Exposing Limitations to HBase Cluster Deployers

Roni Sandel*
Electrical Engineering and Computer Science
York University
Toronto, ON, Canada
rsandel@yorku.ca

Marios Fokaefs
Electrical Engineering and Computer Science
York University
Toronto, ON, Canada
fokaefs@yorku.ca

Mark Shtern
Electrical Engineering and Computer Science
York University
Toronto, ON, Canada
mark@cse.yorku.ca

Hamzeh Khazei
Electrical Engineering and Computer Science
York University
Toronto, ON, Canada
hkh@yorku.ca

Marin Litoiu
Electrical Engineering and Computer Science
York University
Toronto, ON, Canada
mlitoiu@yorku.ca

ABSTRACT

With the advent of sensor networks and portable devices, data has been produced rapidly and in great amount. As a result storing and processing Big Data, in combination with the advances in cloud and virtual infrastructures, pose interesting challenges. In our previous work, we studied these challenges with various experiments around different HBase cluster configurations and their impact on the performance of the cluster. A by-product of our experiments was that, in spite of advances in tooling support to set up and configure a Big Data cluster, the various tools are not always aligned to produce the optimal or near-optimal performance for data clusters. More specifically, we show that the default configuration values of state-of-the-art cluster deployers, including Cloudera, IBM BigInsights, Apache Hortonworks and the manual HBase deployment, do not take in to account the underlying infrastructure resulting in subpar performance.

Keywords

cloud computing; big data; design; performance modelling and evaluation; migration

1. INTRODUCTION

Data has been produced rapidly and in great numbers both within specific domains and from every day activities. However, it is only recently that we have been able to produce and capture this large amount of data efficiently and effectively, mainly thanks to recent technological advances in wireless sensor networks and mobile devices. In turn, this has prompted additional technological advances in the fields of data storing and processing. First, NoSQL databases have been proposed to provide efficient storing of Big Data and fast access. Second, novel algorithms and frameworks, like MapReduce, Hadoop and Spark among others, have been invented for the efficient and accurate processing of Big Data. Moreover, cloud and virtual infrastructure have been leveraged to deploy Big Data solutions in a flexible manner allowing for distributing data and its processing among different physical or virtual machines.

In addition to these technologies, tooling support has been contributed to connect the storage and the processing of the data with the underlying technologies. These tools enable the users to easily set up their data-processing clusters, configure the parameters to store and process the data and choose the best cluster topology in terms of computing, storing and network resources. Although these tools do facilitate the cluster configuration, they act more like graphical "wizards" and do not have the logic to correlate all aspects of the cluster with the goal to promote an optimal configuration under the given conditions.

In our previous work [6], we have encountered such limitations. More specifically, we have conducted experiments to study the impact of decisions concerning the way data is stored and the topology on which the data-processing cluster will be deployed with respect to the performance of the cluster. In a subset of these experiments, we deployed an HBase cluster in various combinations of virtual machines of different numbers and sizes using the Cloudera Hadoop Manager as our deployer. Our intuition was that smaller clusters of large VMs will have better performance than larger clusters of small VMs, due to data locality and more computational power. However, our initial intuition proved to be false. When we further investigated this phenomenon, we realized that the Cloudera Manager deploys a cluster with default parameters regardless of the underlying topology. As a result, the computational power of large VMs was underutilized by the HBase cluster due to suboptimal configuration parame-

*Roni Sandel is currently with IBM Canada, but the work was conducted while at York University.

ters.

In this work, we scrutinize this phenomenon and point out the need for smarter deployers that would take into account all components of a data-processing cluster. First, in Section 2, we summarize the results of our previous experiments with particular focus on how the Cloudera Manager affected the performance of HBase clusters. Second, in Section 3, we provide further proof that this limitation occurs across many deployers other than Cloudera, including IBM BigInsights, Hortonworks, as well as the HBase manual deployment. In Section 4, we propose a tentative solution for this problem based on the concept of training the deployer given the preferred solutions for storing and processing the data and the underlying topology of the cluster. Finally, in Section 5 we discuss a few hand-picked research works that point out the general problem of configuring data-processing clusters and what can go wrong, before Section 6 concludes our work with a discussion about the research potential of the presented challenge.

2. HBASE CLUSTER PERFORMANCE EXPERIMENTS

We previously conducted experiments to assess the impact of various cluster configuration decisions on the performance of HBase clusters [6]. More specifically, we focused on how the HBase schema, the compression of data and the underlying topology of virtual machines affect the response time of the cluster against scan queries by different numbers of users.

Concerning the topology experiments, we imposed a constraint to the available budget for our virtual machines. This limitation resulted in 8 topologies with various combinations of medium, large and extra large VMs. Our original intuition was that smaller clusters with few large VMs will perform better due to the higher computational power of the underlying infrastructure. Figure 1 shows the response time for three representative topologies against various number of users; one with 8 medium VMs (e2), one with 2 large and 1 extra large VMs (e3) and one with 2 medium VMs and 3 large VMs (e6). The figure clearly shows that our intuition was wrong and that the cluster with the 8 medium VMs outperforms the others by a large margin.

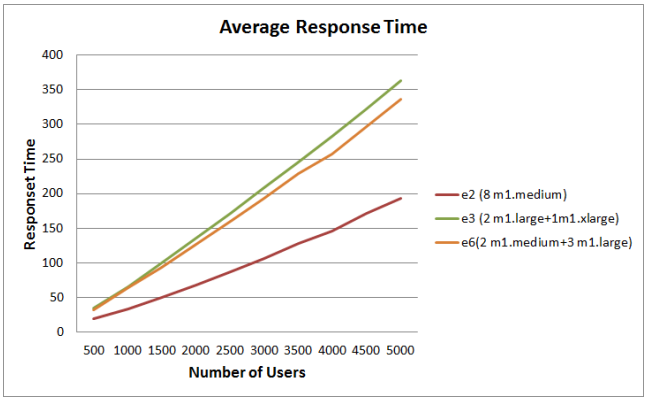

Figure 1: Average response time for different topologies

We decided to further investigate this presumed anomaly. The reason behind it lies in the default values for the deployment of the cluster by the Cloudera Manager. By default, the deployer assigns 500MB of memory to the Java heapsize for medium instances, 850MB for large instances and 700MB for extra large instances. Given the memory specifications for the various sizes of VM (3.75GB for medium, 7.5GB for large and 15GB for extra large), this default assignment obviously underutilizes the VM's memory and this effect is more prominent as the size increases, since the deployer does not take into account this parameter. The justification of the difference in performance is load balancing; in large clusters the workload is highly distributed among VMs, while in smaller clusters is more condensed. Given the fact that VMs in both clusters are assigned the same capacity, the workload is exaggerated in the large VMs of the smaller cluster. In principle, if we perceive the cluster as a single processing node, the cluster of e2 has totally 8GB heapsize, while in e6 the cluster has 5GB heapsize. It is, then, naturally anticipated that e2 would perform faster than the other clusters.

3. DEFAULT VALUE LIMITATION OF CLUSTER DEPLOYERS

Having identified the limitation in the default settings of the Cloudera Manager and having proven that this affects negatively the performance of the HBase cluster, we further investigate whether this is an isolated case or if other state-of-the-art deployers behave similarly. Our suspicion of this being the general case was confirmed after we looked into the documentation of four more deployers; IBM BigInsights [3], Hortonworks Data Platform [2], which uses Apache Ambari as its deployer[1] and HBase itself [1], when deployed by itself without the use of a deployer. All these frameworks use 1GB for the heapsize regardless of the VM size.

IBM BigInsights, in its documentation [3], recognizes that 1GB for the heapsize is very small for an intensively used HBase cluster and recommends increasing the default size as much as possible. In fact, in the provided example, the default value is raised to 8GB. Hortonworks in particular has the ability to recognize the size of the VMs for the worker nodes and the services that will be deployed in them. Additionally, in its documentation, there is discussion on how to distribute the memory of the VM for each service. Nevertheless, even with this knowledge, it does not calculate a better default parameter for the heapsize to suit the capacity of the underlying topology.

All these deployers use the 1GB of heapsize as this is a minimum value to guarantee the proper function of the cluster. On one hand, this size is less than what is usually the minimum size of RAM in a VM offered by public clouds, like Amazon[2] or Rackspace[3]. On the other hand, the deployers may assume that the HBase services, including MapReduce, TaskTrackers and so on, may be deployed along with other services that will reduce the available memory. However, the fact that cloud infrastructures allow for the creation of dedicated clusters that can be easily commissioned and decommissioned at will and that these services are now more accessible to moderately or even minimally experienced users,

[1] `https://ambari.apache.org/` Last accessed: 21-Jun-2015
[2] `http://aws.amazon.com/ec2/instance-types/` Last accessed: 21-Jun-2015
[3] `http://www.rackspace.com/cloud/servers` Last accessed: 21-Jun-2015

accentuates the need for smarter deployment tools that will also be able to guarantee better performance from the clusters.

4. BOOTSTRAPPING THE DEPLOYMENT OF HBASE CLUSTERS

To achieve smarter deployment, we argue that two ingredients are necessary; additional knowledge about the capacity of the underlying (physical or virtual) infrastructure and additional functionality on the deployer's part to parse this extra knowledge and recommend such parameter values to ensure better performance than the one guaranteed by the default ones.

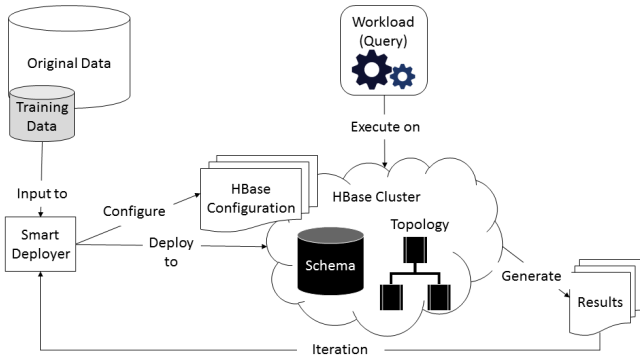

Figure 2: Bootstrapping the cluster to improve its configuration.

It is important to note that no more accurate values can be acquired unless the infrastructure and the data-processing cluster are tested with actual data. To this end, we tentatively propose a bootstrapping method to train deployers as shown in Figure 2. Initially, a small portion of the actual data, randomly sampled, is dedicated to the training of the deployer. Next the cluster is set up with the underlying topology and the proper schema. We assume that the best alternatives for these properties have already been found, possibly using the method from our previous work [6]. The data is then transferred to the cluster, which is exercised using synthetic workload, but representative of the actual use cases for the cluster. In principle, it is preferred to have workload that would stress the cluster to its performance limits (e.g. "scan" queries), in order to acquire the most accurate values for the configuration parameters.

After the cluster has been exercised a few times (to account for the statistical error), the real values for certain parameters are gathered by the deployer. Nominal or face values of parameters as these are presented by the cloud provider (e.g. RAM size or network bandwidth) do not always correspond to the actual values for a VM and they affect the performance of the cluster if they are used as they are. Additionally, exercising the cluster enables the measurement of a multitude of other parameters including, but not limited to, disk I/O speed, processor clock speed, network bandwidth and throughput and how RAM is allocated and utilized by all the services that may be deployed in a VM. Furthermore, these measurements allow us to get a complete picture of each VM, especially in the case of heterogeneous clusters.

When relevant metrics have been gathered, they are processed by the deployer to find the best values for the cluster configuration parameters and then redeploys the cluster. The cycle is repeated, either for a fixed number of times or until the configuration values converge. We assume, although it still remains to be confirmed, that a small training period on a small portion of the data would suffice to produce good enough configuration parameters, and considerably better than the default values, to ensure at least better utilization of the underlying infrastructure. The proposed method does not guarantee optimal configuration, as this would require a more complicated and thorough process. Our method is simple enough for the improved performance it can result in. Given the size of modern data-processing tasks and applications, we believe that the training period can have an insignificant overhead to the whole process, especially considering the potential improvement.

5. RELATED WORK

Rabkin and Katz [4] present an elaborate discussion on the points of possible failure for data-processing clusters and more particularly for Hadoop clusters. Relative to our work, they found that the biggest source of problems in the cluster was actually misconfiguration of the various parts. Specifically for HBase, misconfigurations accounted for about 30% of the total errors for this software. Out of all the possible misconfigurations, RAM and thread allocation were the most frequently occurring ones.

Rao *et al.* [5] also discuss Hadoop clusters, but in the context of potential performance issues in heterogeneous clusters. They provide general guidelines and heuristics on how to properly set up and configure the cluster. They split their guidelines in three broad categories; hardware, application and how to avoid system bottlenecks. They also recommend benchmarking as a method to find the best hardware topology and they give heuristics on how to tune the number of tasks and replication in the cluster. Their guidelines are mostly theoretic derived from the service documentation and they do not go into details on how to configure the services for a given topology or a given set of tasks.

Shtern *et al.* [7] discuss a variety of interesting research issues that arose from the work in the context of a multi-cluster analytical engine. The authors identify the configuration and deployment of a data-processing cluster on the cloud as a source of potential problems. This challenge may also be exaggerated in the case of multiple clusters, since the user will have to deal with a higher degree of distribution, more than one stakeholders and most likely uneven distribution of permissions and privileges. Another interesting challenge identified by the authors is the ability for the cluster to be elastic and automatically scale up or down. In the context of our work, this property makes the dynamic and automatic identification of better default values for configuration parameters imperative.

6. CONCLUSION

In this paper, we point out a limitation in state-of-the-art data-processing cluster deployers. The limitation concerns the default values for the configuration of the Big Data clusters. We originally suspected this limitation after we conducted a series of experiments concerning the evaluation of various cluster configurations. In this experiments, we used

Cloudera Manager as the deployer and we found that it used rather conservative default values for the configuration parameters, which led to suboptimal and counterintuitive performance results. We confirmed that this is not an isolated limitation but it spans across many state-of-the-art deployers including, IBM BigInsights, Hortonworks and it was also identified in the documentation for the manual deployment of a HBase cluster.

The choice of such moderate default values can be justified as it covers the minimum expectations for the infrastructure of a data-processing cluster and possibly the level of the user's expertise. However, it is our argument that the deployers have to become smarter in assigning these values in order to keep up with the rest of the tooling support for data clusters. On one hand, cloud infrastructure are becoming even more flexible and can be shaped directly for the purposes of a data cluster. On the other hand, thanks to tooling support and interactive management systems, data-processing systems are becoming more accessible to users regardless of their expertise level. In light of this, users may require more guidance towards fine-tuning the cluster. Currently, users may optimize the performance of the cluster either empirically, or through trial and error, or using some expert advice (for example, from deployer or data storage documentation).

In addition to pointing out the limitation, in this paper, we also initialize the proposition for a smarter deployer. We propose to train the deployer using a portion of the actual data to be processed in order to enable definition of optimal values for the configuration parameters by taking into account the underlying infrastructure. Our method is still in an embryonic state, but we plan to further develop and eventually extend it into a tool or incorporate it in an existing deployer or cloud management system.

Nevertheless, we strongly believe that the discussed problem constitutes a significantly interesting research opportunity. The results of such research can help not only to extend existing tooling and management support but also to make data-processing even more accessible and further facilitate users in deploying their tasks regardless of their expertise or technical knowledge about the whole spectrum of infrastructures required by the cluster. In any case, any contribution will help enhance recommendations that could be overridden manually and, as such, we expect very little negative overhead to the users. Finally, any contribution towards this line of research will increase the degree of automation and reduce the required time and effort as well as minimize the margin for error or subpar performance.

Acknowledgments

This research was supported by IBM Centres for Advanced Studies (CAS), the Natural Sciences and Engineering Council of Canada (NSERC) under the Smart Applications on Virtual Infrastructure (SAVI) Research Network, and the Ontario Research Fund for Research Excellence under the Connected Vehicles and Smart Transportation (CVST) project.

7. REFERENCES

[1] Apache HBase âĎć Reference Guide - hbase-env.sh. `https://hbase.apache.org/book.html#hbase_env`. Last accessed: 24-Jun-2015.

[2] Hortonworks Data Platform - Recommended Memory Configurations for the MapReduce Service. `http://docs.hortonworks.com/HDPDocuments/Ambari-1.6.1.0/bk_ambari_reference/content/mem-configs-mapReduce.html`. Last accessed: 24-Jun-2015.

[3] IBM InfoSphere BigInsights Version 3.0 - General HBase tuning. `https://www-01.ibm.com/support/knowledgecenter/SSPT3X_3.0.0/com.ibm.swg.im.infosphere.biginsights.analyze.doc/doc/bigsql_gentune.html`. Last accessed: 24-Jun-2015.

[4] A. Rabkin and R. H. Katz. How hadoop clusters break. *Software, IEEE*, 30(4):88–94, 2013.

[5] B. T. Rao, N. Sridevi, V. K. Reddy, and L. Reddy. Performance issues of heterogeneous hadoop clusters in cloud computing. *arXiv preprint arXiv:1207.0894*, 2012.

[6] R. Sandel, M. Shtern, M. Fokaefs, and M. Litoiu. Evaluating cluster configurations for big data processing: An exploratory study. In *2015 Symposium on the Maintenance and Evolution of Service-Oriented Systems and Cloud-Based Environments*. IEEE, 2015. (accepted).

[7] M. Shtern, R. Mian, M. Litoiu, S. Zareian, H. Abdelgawad, and A. Tizghadam. Towards a multi-cluster analytical engine for transportation data. In *Cloud and Autonomic Computing (ICCAC), 2014 International Conference on*, pages 249–257. IEEE, 2014.