International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) e-ISSN: 2395-5325
Volume3, Issue 1 (January'2017)

PROFITABLE RESOURCE ALLOCATION USING MAP REDUCE WITH CURA TECHNIQUES

Dr.S.Suriya,
Professor,
Department of Computer Science and Engineering,
K.L.N College of Information Technology,
Sivaganga,Tamilnadu,India.

K.Nagalakshmi
Associate Professor,
Department of Computer Science and Engineering,
K.L.N College of Information Technology,
Sivaganga,Tamilnadu,India.

G.Balakrishnan,
Associate Professor,
Department of Computer Science and Engineering,
K.L.N College of Information Technology,
Sivaganga,Tamilnadu,India.

S.J.Subhashini
Associate Professor,
Department of Computer Science and Engineering,
K.L.N College of Information Technology,
Sivaganga,Tamilnadu,India.

Abstract: This paper presents a new MapReduce cloud service model, Cura, for profitable MapReduce services in a cloud. In contrast to existing MapReduce cloud services such as a generic compute cloud or a dedicated MapReduce cloud, Cura has a number of unique benefits. First, Cura is designed to provide a cost-effective solution to efficiently handle MapReduce production workloads that have an important amount of interactive jobs. Second, unlike existing services that require customers to decide the resources to be used for the jobs, Cura leverages MapReduce profiling to automatically create the best cluster configuration for the jobs. While the existing models allow only a per-job resource optimization for the jobs, Cura implements a globally efficient resource allocation scheme that significantly reduces the resource usage cost in the cloud. Third, Cura leverages unique optimization opportunities when dealing with workloads that can withstand some slack .By effectively multiplexing the available cloud resources among the jobs based on the job requirements, Cura achieves significantly lower resource usage costs for the jobs and it also includes identifying the shortest execution time. Cura's core resource management schemes include cost-aware resource provisioning, M-aware scheduling and online virtual machine reconfiguration. Our experimental results using Census workload traces show that our techniques lead to more than 80 percent reduction in the cloud compute infrastructure cost with upto 65 percent reduction in job response times.

Keywords: Mapreduce, Cloud Services, bigdata, Hadoop

I. INTRODUCTION

Cloud computing and its pay-as-you-go cost structure have enabled hardware infrastructure service providers platform service providers as well as software and application service providers to offer computing services on demand and pay per use just like how we use utility today. This growing trend in cloud computing, combined with the demands for Big Data and Big Data analytics, is driving the rapid evolution of datacenter technologies towards more profitable, consumer-driven and technology agnostic solutions. The most popular approach towards such big data analytics is using Map Reduce and its open-source implementation called Hadoop. Offered in the cloud, aMapReduce service allows enterprises to analyze their data without dealing with the complexity of building and managing large installations of Map Reduce platforms. Using virtual machines and storage hosted by the cloud, enterprises can simply create virtual Map Reduce clusters to analyze their data. In this paper, we discuss the cost-inefficiencies of the existing cloud services for MapReduce and propose a profitable resource management framework called Cura that aims at a globally optimized resource allocation to minimize the infrastructure cost in the cloud datacenter. We note that the existing cloud solutions for Map Reduce work primarily based on a per-job or per-customer optimization approach where the optimization and resource sharing opportunities are restricted within a single job or a single customer. Here there source optimization opportunity is restricted to the perjoblevel. Alternately, one can lease dedicated cluster resources from a generic cloud service like Amazon Elastic Compute Cloud and operate MapReduce on them as if they were using a private MapReduce infrastructure. While this approach enables resource optimization at the per-customer level. Cura on the other hand is designed to provide a cost-effective solution to a wide range of MapReduce workloads with the following goals in mind: First, we observe that existing solutions are not costeffectiveto deal with interactive MapReduce workloads that consist of a significant fraction of short running jobs with lower latency requirements. A recent study on the Census production workload traces reveals that more than 95 percent of their production MapReduce jobs are short running jobs with an average running time of 30 secs. Unlike existing per-job services that require VMs to be created afresh for each submitted job, Cura deals with such interactive workloads using a secureinstant VM allocation scheme that minimizes the job.

II.RELATED WORK

Resource allocation and job scheduling. There is a large body of work on resource allocation and job scheduling in grid and parallel computing. Some representative examples of generic

schedulers include [37], [38]. The techniques pro-posed in [39], [40] consider the class of malleable jobs where the number processors provisioned can be varied at run-time. Similarly, the scheduling techniques presented in [41], [42] consider moldable jobs that can be run on different number of processors. These techniques do not consider a virtualized setting and hence do not deal with the challenges of dynamically managing and reconfiguring the VM pools to adapt for workload changes. Therefore, unlike Cura they do not make scheduling decisions over dynamically managed VM pools. Chard et al. present a resource allocation framework for grid and cloud computing frame-works by employing economic principles in job scheduling [45]. Hacker and Mahadik propose techniques for allocating virtual clusters by queuing job requests to minimize the spare resources in the cloud [46]. Recently, there has been work on cloud auto scaling with the goal of minimizing customer cost while provisioning the resources required to pro-vide the needed service quality [43]. The authors in [44] propose techniques for combining on demand provisioning of virtual resources with batch processing to increase sys-tem utilization. Although the above mentioned systems have considered cost reduction as a primary objective of resource management, these systems are based on either per-job or per-customer optimization and hence unlike Cura, they do not lead to a globally optimal resource management. MapReduce task placement. There have been several efforts that investigate task placement techniques for MapReduce while considering fairness constraints [17], [32]. Mantri tries to improve job performance by minimizing outliers by making network-aware task placement [3]. Similar to Yahoo's capacity scheduler and Facebook's fairness scheduler, the goal of these techniques is to appropriately place tasks for the jobs running in a given Hadoop cluster to optimize for locality, fairness and performance. Cura, on the other hand deals with the challenges of appropriately provisioning the right Hadoop clusters for the jobs in terms of VM instance type and cluster size to globally optimize for resource cost while dynamically reconfiguring the VM pools to adapt for workload changes. MapReduce in a cloud. Recently, motivated by MapRe-duce, there has been work on resource allocation for data intensive applications in the cloud context [18], [33]. Quincy [18] is a resource allocation system for scheduling concur-rent jobs on clusters and Purlieus [33] is a MapReduce cloud system that improves job performance through locality opti-mizations achieved by optimizing data and compute place-ments in an integrated fashion. However, unlike Cura these systems are not aimed at improving the usage model for MapReduce in a cloud to better serve modern workloads with lower cost.

III.CURA: MODEL AND ARCHITECTURE

In this section, we present the cloud service model and system architecture for Cura.

a. Cloud Operational Model

The first operational model is a completely customer managed model where each job and its resources are specified by the customer on a per-job basis and the cloud provider only ensures that the requested resources are provisioned upon job arrival. Many existing cloud services such as Amazon Elastic Compute Cloud use this model. This model has the lowest rewards since there is lack of global optimization across jobs as well as other drawbacks discussed earlier. The second possible model (delayed start) is partly customer-managed and partly cloud-managed model where customers specify which resources to use for their jobs and the cloud provider has the flexibility to schedule the jobs as long as they begin execution within a specified deadline. Here, the cloud provider takes slightly greater risk to make sure that all jobs begin execution within their deadlines and as a reward can potentially do better multiplexing of its resources. However, specifically with MapReduce, this model still provides low cost benefits since jobs are being optimized on a per-job basis by disparate users. In fact customers in this model always tend to greedily choose low-cost small cluster configurations involving fewer VMs that would require the job to begin execution almost immediately.

For example, consider job that takes 180 minutes to complete in a cluster of two small instances but takes 20 minutes to complete using a cluster of six large instances.1 Here if the job needs to be completed in more than 180 minutes, the per-job optimization by the customer will tend to choose the cluster of two small instances as it has lower resource usage cost compared to the six large instance cluster. This cluster configuration, however, expects the job to be started immediately and does not provide opportunity for delayed start. This observation leads us to the next model. The third model—which is the subject of this paper—is a completely cloud managed model where the customers only submit jobs and specify job completion deadlines. Here, the cloud provider takes greater risk and performs a globally optimized resource management to meet the job SLAs for the customers. Typically, the additional risks here include the responsibilities of meeting additional SLA requirements such as executing each job within its deadline and managing the allocation of resources for each job. While the conventional customer-optimized cloud model requires only VMs to be provisioned based on demand, a completely cloud managed model introduces additional role on the cloud provider for resource management. For instance, an inefficient allocation of resources to a particular job can result in higher cost for the cloud provider. Therefore, this model brings higher risk to the cloud while it has high potential cost benefits. Similar high-risk high-reward model is the database-as a service model, where the cloud provider estimates the execution time of the customer queries and performs resource provisioning and scheduling to ensure that the queries meet their response time requirements. As MapReduce also lends itself well to prediction of execution time, we have designed Cura on a similar model. Another recent example of this model is the Batch query model in Google's Big Query cloud service where the cloud provider manages the resources required for the SQL-like queries so as to provide a service level agreement of executing the query within 3 hours.

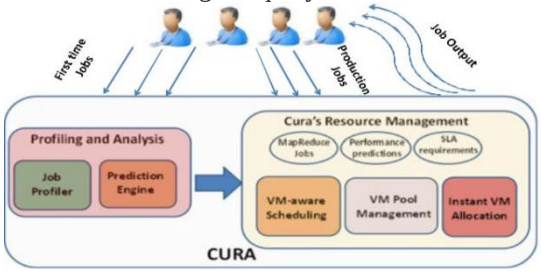


Figure 1: Cura System Architecture.

b. System Architecture

The profile and analyze service is used only once when a customer's job first goes from development-and-testing into production in its software life cycle. For subsequent instances of the production job, Cura directly sends the job for scheduling. Since typically production jobs including interactive or long running jobs do not change frequently profiling will most often be a one-time cost. Further, from an architectural standpoint, Cura users may even choose to skip profiling and instead provide VM type, cluster size and job parameters to the cloud service similar to existing dedicated MapReduce cloud service models. Jobs that skip the one-time profile and analyze step will still benefit from the response time optimizations in Cura described below, however, they will fail to leverage the benefits provided by Cura's global resource optimization strategies. Jobs that are already profiled are directly submitted to the Cura resource management system.

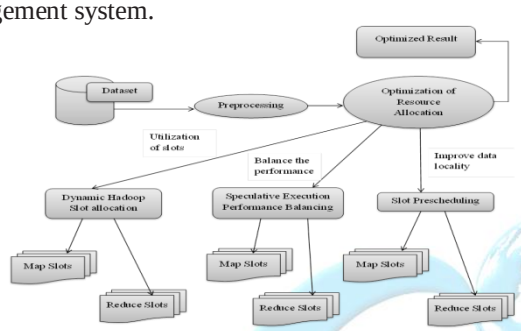


Figure 2: System Architecture.

IV.IMPLEMENTATION

a). Dataset Upload: When a user intends to access the file, the sender sends his information to be authenticated by the data owner. In our design, we use our unified school authentication in cost and transfer it through https for safety concern .The data owner sends the keys along with the hash table back if the user belongs to the legal set. This hash table will be used in the hash process. Then the data owner records the" International Mobile Equipment Identity" of the user's mobile device and stores its encrypted version into the cloud.

b). Dataset Preprocessing: Imputation is the process of replacing missing data with values. Single imputation-A once-common method of imputation was hot-deck imputation where a missing value was imputed from a randomly selected similar record. In cases with imputation, there is guaranteed to be no relationship between the imputed variable and any other measured variables.

c). HDFS Upload: Hadoop Distributed File System (DFS) will be configured for uploading the preprocessed geo data into hadoop. The configuration includes setting VM (hadoop platform) IP and port for connection. FSDataInputStream and FSDataOutputStream is used to upload and download data from hadoop. Different datacenters are analyzed for data execution across different datacenters.

d). Slot Allocation : Here we are going to allocate the slot based on dynamic Hadoop slot allocation optimization mechanism. In the slot pre-scheduling process we are going to improve the data locality. Slot Pre-Scheduling technique that can improve the data locality while having no negative impact on the fairness of Map-Reduce jobs. Some idle slots which cannot be allocated due to the load balancing constrain during runtime, we can pre-allocate those slots of the node to jobs to maximize the data locality.

e). VM Server: In contrast to existing Map Reduce services that create VMs on demand, Cura employs a secure instant VM allocation scheme that reduces response times for jobs, especially significant for short running jobs. Upon completion of a job's execution, Cura only destroys the Hadoop instance used by the job (including all local data) but retains the VM to be used for other jobs that need the same VM configuration.

f. Performance Evaluation : Speculative execution is a common approach for dealing with the straggler problem by simply backing up those slow running tasks on alternative machines. Multiple speculative execution strategies have been proposed, but there is a pitfall: incoming jobs are allocated to nodes present in server and fail to schedule process type allocate to node for processing. Performance is evaluated by means of selective the optimized resources and results taken in terms of execution time, processing memory etc.

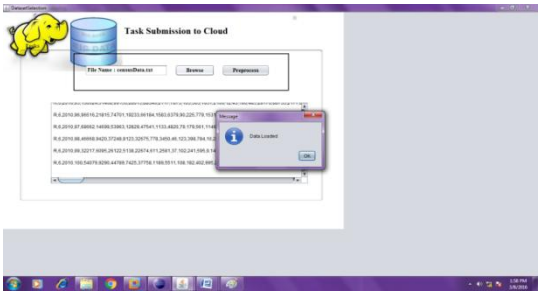
V. EXPERIMENTAL EVALUATION

We divide the experimental evaluation of Cura into two—first, we provide detailed analysis on the effectiveness of Cura compared to conventional MapReduce services and then we present an extensive micro analysis on the different set of techniques in Cura that contribute to the overall performance. We first start with our experimental setup.

Experimental Results: We first present the experimental evaluation of Cura by comparing with the existing techniques for various experimental conditions determined by distribution of the job deadlines, size of the MapReduce jobs, number of servers in the system and the amount of prediction error in the profile and analyze process. By default, we use a composite work-load consisting of equal proportion of jobs of three different categories: small jobs, medium jobs and large jobs. Small jobs read 100 MB of data, whereas medium jobs and large jobs read 1 and 10 GB of input data respectively. We model Poisson job arrivals with rate parameter, ¼ 0:5 and the jobs are uniformly distributed among 50 customers. The evaluation uses 11,500 jobs arriving within a period of 100 minutes. Each of the arrived job represents one of the 50 profiled jobs with input data size ranging from 100 MB to 10 GB based on the job size category. By default, we assume that jobs run for the same amount of time predicted in the profile and analyze process, however, we dedicate a separate set of experiments to study the performance of the techniques when such predictions are erroneous. Note that a job's complete execution includes both the data loading time from the storage infrastructure to the compute infrastructure and the Hadoop start up time for setting up the Hadoop cluster in the cluster of VMs. The data loading time is com-puted by assuming a network throughput

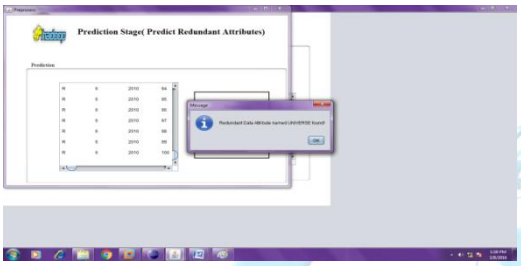
of 50 MBps per VM[5] from the storage server and the Hadoop startup time is taken as 10 sec.

5.1.1 DATA SET UPLOAD

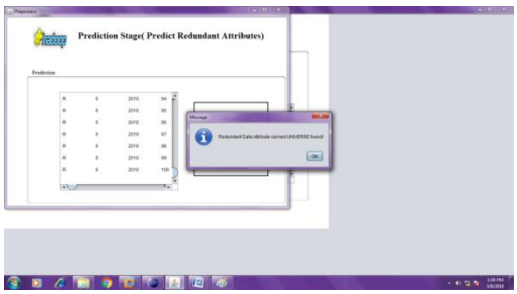


Here we are uploading the data's . The data owner sends the keys along with the hash table back if the user belongs to the legal set. This hash table will be used in the hash process .Then the data owner records the" International Mobile Equipment Identity" of the user's mobile device and stores its encrypted version into the cloud.

5.1.2 DATASET PREPROCESSING

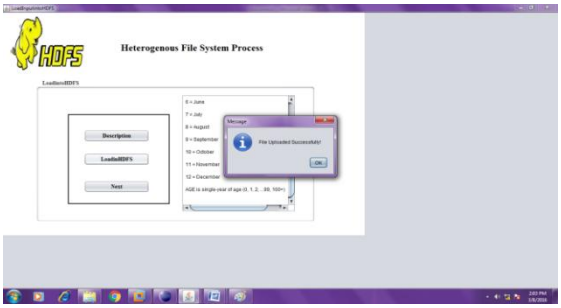


Here we use a technique called "imputation" ,which replaces the misplaced words. There is single imputation and many forms. Data preprocessing also includes removal of unwanted punctuations and symbols. It removes the letter which exhists in more than one times.



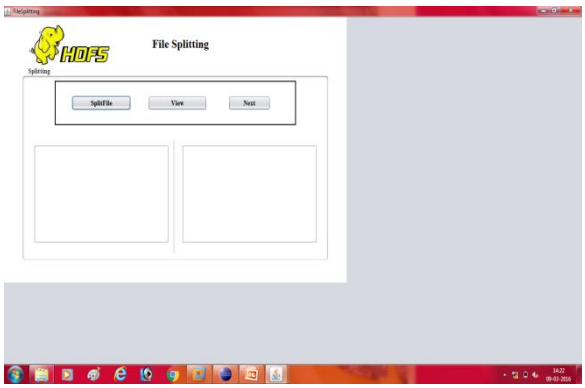
Now the data is preprocessed.

5.1.3. HDFS UPLOAD



The data's are uploaded in HDFS. Hadoop Distributed File System (DFS) will be configured for uploading the preprocessed geo data into hadoop.

5.1.4 SLOT ALLOCATION



Slots are allocated on their respective groups. Here we are going to allocate the slot based on dynamic Hadoop slot allocation optimization mechanism .Slot Pre-Scheduling technique that can improve the data locality while having no negative impact on the fairness of Map-Reduce jobs.
Next connecting the VM Server and evaluating their performance.

VI. CONCLUSION

This paper presents a new MapReduce cloud service model, Cura, for data analytics in the cloud. We argued that existing cloud services for MapReduce are inadequate and inefficient for production workloads. In contrast to existing services, Cura automatically creates the best cluster configuration for the jobs using MapReduce profiling and leverages deadline-awareness which, by delaying execution of certain jobs, allows the cloud provider to optimize its global resource allocation efficiently and reduce its costs. Cura also uses a unique secure instant VM allocation technique that ensures fast response time guarantees for short interactive jobs, a significant proportion of modern MapReduce work-loads. Cura's resource management techniques include cost-aware resource provisioning, VM-aware scheduling and online virtual machine reconfiguration. Our experimental results using jobs profiled from realistic Facebook-like production workload traces show that Cura achieves more than 80 percent reduction in infrastructure cost with 65 per-cent lower job response times.

VII.REFERENCES

[1]. B.Palanisamy,A.Singh,L,Liu, " Cost Effective Resource Provisioning for MapReduce in a Cloud", IEEE Trans.Parallel Distrib.System., vol.26 ,no.5, pp.1265-1279,May-2015.

[2]. D.Loreti, A.Ciampolini, "A Hybrid Cloud Infrastructure for Big Data Applications", in Proc.Conf.hot Topics Cloud Comput, 2015.

[3]. T.Hacker and K..Mahadik, "Flexible resource allocation for reliable virtual cluster computing systems," in Proc. Int. Conf. High Perform. Comput, Netw., Storage Anal,2014 pp.1-12.

[4]. Z.Xiao, W.Song, Q.Chen, "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment ",IEEE Trans.Parallel Distrib.System.,vol.24,no.6,pp.1107-1117,June-2013.

[5]. J.Polo, C.Castillo, D.Carrera, "Resource aware Adaptive scheduling for MapReduce Clusters", cited by 88 , 2013.