CLOUDLET: Towards MapReduce Implementation on Virtual Machines

Shadi Ibrahim, Hai Jin, Bin Cheng, Haijun Cao, Song Wu
Cluster and Grid Computing Lab
Services Computing Technology and System Lab
Huazhong University of Science and Technology
Wuhan, 430074, China
{shadi, hjin}@hust.edu.cn

Li Qi
Operation Center
China Development Bank
Beijing, China
quick.qi@gmail.com

ABSTRACT
The existing *MapReduce* framework in virtualized environment suffers from poor performance, due to the heavy overhead of I/O virtualization, and management difficulty for storage and computation. To address the problems, we propose **Cloudlet**, a novel *MapReduce* framework on virtual machines. The aim of **Cloudlet** design is to overcome the overhead of VM while benefiting of the other features of VM (i.e. management and reliability issues).

Categories and Subject Descriptors
C.1.2 **[Processor Architectures]:** Multiple Data Stream Architectures—*Parallel processors*; D.4.7 **[Operating System]**: Organization and Design— *Distributed Systems*.

General Terms
Design, Management, Performance.

Keywords
Data Intensive, MapReduce, Hadoop, Virtual Machine.

1. INTRODUCTION
As the size and complexity of modern data intensive computing system are rapidly increasing to meet the demands of the extreme inflation of data volumes generated by science, humanities and entitlements, especially after the widely acceptance and increasing popularity of cloud computing services, (e.g. Storage as a Service), reliability, manageability and high performance of computing infrastructure are becoming critical concerns to achieve high elastic on-demand services to meet the user's requirements. On the other hand, *MapReduce* [4], due to its magnificent features including simplicity, fault tolerance, and scalability has been emerging as an important programming model for large scale data parallel applications such as web indexing, data mining, and scientific simulation. Meanwhile, due to the benefits of ease system management, security, resource consolidation and power management, virtual machine (**VM**) technologies have become popular in both industry and academia. Recently, with the increasing maturity of virtualization technology, virtual machines have been a promising approach for HPC systems [7]. However, combining VMs and *MapReduce* for data intensive computing systems, (e.g. Amazon has added *Hadoop* [1] to their *Amazon Machine Image* (**AMI**) applications stack, to provide large distributed processing *Hadoop* cluster for data intensive applications as in *Alexa* [2]), is still claimed to be inefficient [5, 6], due to the heavy overhead of I/O virtualization [8]. The reasons for the unacceptable overhead of *MapReduce*, *Hadoop* in practice, on VMs are: a) the performance of DFS on VM is not good enough, b) VMs are competing for disk and network bandwidth, resulting in slow data transfer among VMs during the copy phase of the reduce task execution (as shown in Figure 1). It indeed leads to unfair and massive executions of speculative (e.g. 80% of tasks are speculatively executed [5]) and incurs the large amount of network overhead.

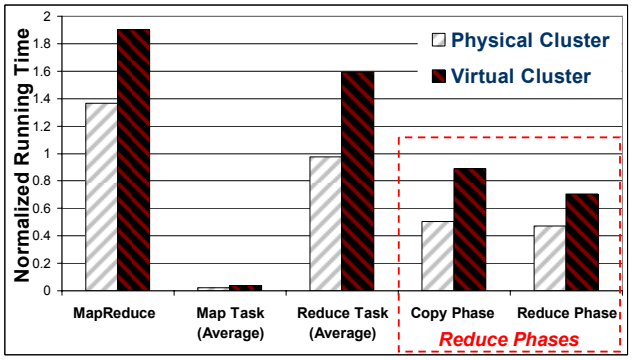

Figure 1. *MapReduce* on Physical Cluster Vs Virtual Cluster. We conducted our performance evaluations on 7 nodes cluster. Each node is equipped with two 4-core 2.33GHz Xeon processors, 8GB of memory and 1TB of disk, runs RHEL5 with kernel 2.6.22, and is connected with 1GB Ethernet. In VM-based environments, we use Xen 3.2. VMs are running with REHEL 5, kernel 2.6.22. In both cases, we have 1 master and 6 slaves. Each VM (2VCPU, 2GB Mem) is uniquely deployed on physical machine. The results are obtained with *Hadoop* (0.18.0) using sort job on 12GB of data.

To address the above problems, we propose a novel *MapReduce* framework on virtual machines called **Cloudlet**. Our new design is mainly based on the following concerns: a) it is feasible to fully utilize the physical node resources, using VM only as a computation unit for the data located on its physical node, b) as VMs are highly prone to error, it is important to separate the permanent data storage (DFS) from the virtual storage associated with VM, c) it is useful to bound the data transfer to/from the VMs within the physical host machine, and d) it is useful to decrease the amount of outer data transfer (among data nodes), by

HPDC'09, June 11-13, 2009, Munich, Germany.
ACM 978-1-60558-587-1/09/06.

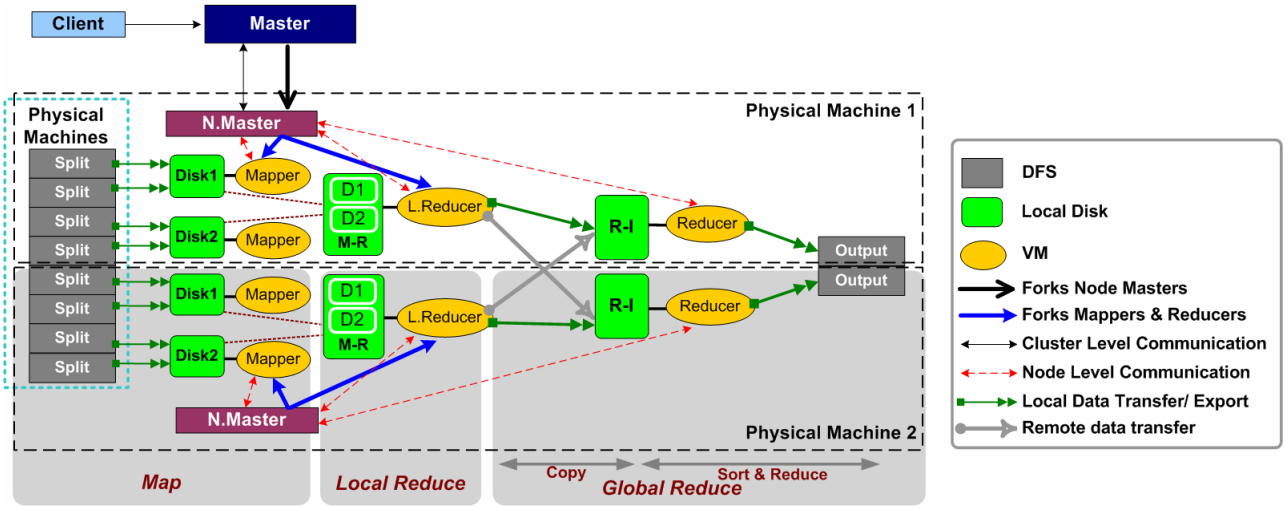

Figure 2. Cloudlet Execution Overview

starting the local reduce tasks after all the map tasks are successfully executed.

2. CLOUDLET

Cloudlet design is simply by resembling *MapReduce* framework with special emphasizes on the virtual machine based environments.

2.1 Execution Overview

As shown in Figure 2, the execution in **Cloudlet** consists of three main phases: **1) Map**. It starts with executing the maps locally in the VM after exporting the data from the DFS into its virtual disk benefiting of the advanced research in [3]. The output of all the maps executed on the same physical machine will then be mounted to one virtual machine (*L.Reducer* mounts multi-mapper's disk). We call this virtual temporal repository *Map Repository* (M-R). **2) Local Reduce.** Every *L.Reducer* will then execute sort and reduce functions within the same physical node. **3) Global Reduce.** A partition function will be executed on all the *L.Reduces* outputs and perform, under the supervision of the *Master* and the *Node Master* (N.Master), locality-aware global reduce (one reducer per physical machine). Furthermore, after the copy phase is finished all the reducers input will be collected in *Reduce Input* repository (R-I) then we mount R-I to a VM, so sort and reduce function can be executed, with the results being saved to DFS.

2.2 Cloudlet Design Principles

Data replications and redundant execution are used to alleviate the effects of slow machines and machine failure. In addition, VM live migration and checkpointing can be employed to increase the system reliability and to achieve high throughput and execution accurateness. Moreover, due to the limitations of network bandwidth, especially in VM based cluster, several optimizations in **Cloudlet** are therefore proposed to reduce the amount of outer data transfer (i.e. the amount of data sent across the network): 1) execute the reduces after all maps have been finished, 2) execute local reduces within the physical node, and finally 3) locality-aware global reduce execution.

3. FUTURE WORK

Current and future research efforts include developing our *MapReduce* implementation on VMs and experimenting with scheduling and migrating the VMs within a cluster to improve high performance, reliability, manageability and power management.

4. ACKNOWLEDGMENTS

This work is supported by National 973 Key Basic Research Program under grant No.2007CB310900, National Science Foundation of China under grant No.60673174, and Program for New Century Excellent Talents in University under grant NCET-07-0334.

5. REFERENCES

[1] Hadoop: http://lucene.apache.org/hadoop. 2008.

[2] Alexa Web Information Service: http://aws.amazon.com/solutions/case-studies/alexa/. 2008.

[3] XenSource: http://www.xensource.com/. 2008.

[4] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Proceedings of OSDI 2004.*

[5] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica. Improving mapreduce performance in heterogeneous environments. *Proceedings of OSDI 2008.*

[6] S. Ibrahim, H. Jin, and L. Lu. Experiences with MapReduce on VMs. Technical Report. Huazhong University of Science and Technology. December 2008.

[7] W. Huang, J. Liu, B. Abali, and D. K. Panda. A Case for High Performance Computing with Virtual Machines. *Proceedings of ICS 2006.*

[8] A. Menon, J. R. Santos, Y. Turner, G. J. Janakiraman, and W. Zwaenepoel. Diagnosing Performance Overheads in the Xen Virtual Machine Environment. *Proceedings of VEE 2005.*