Replication of Enormous Data Process in Cloud Computing

K.Aishwarya,
Assistant Professor/Dept of CSE,
Arulmigu Meenakshi Amman College of Engineering,
Thiruvannamalai Dt,Near Kanchipuram

R.Sathyan,
Department of CSE,
Arulmigu Meenakshi Amman College of Engineering,
Thiruvannamalai Dt,Near Kanchipuram

***Abstract*—Cloud computing is a computing platform with the back bone of internet to store, access the data and application which is in the cloud, not in the computer. The continuous increase of computational power has produced an overwhelming flow of data. The recent trends in Web technology has made it easy for any user to provide and consume content of any form. Replication in the cloud computing architecture and large scale data processing mechanisms. When replicated data are concurrently updated at different sites or when the system workload and the resources requested by clients change dynamically. In replication location of this infrastructure to the network to reduce the costs associated with the management of hardware and software resources. In this paper, replication of Data Management in the Cloud Computing, data integrity and user privacy through cloud system.**

Keywords*—Cloud Computing, Cloud Data Storage, database management, Replication, MapReduce, Virtual Machine.*

1. Introduction

In the continuous increase of computational power has produced an overwhelming flow of data which called for a paradigm shift in the computing architecture and large scale data processing mechanisms. The first three paradigms in the experimental, theoretical and, more recently, computational science. Gray argued that the only way to cope with this paradigm is to develop a new generation of computing tools to manage, visualize and analyze the data flood. Jim Gray, a database software pioneer and a Microsoft researcher, called the shift a "*fourth paradigm*" [1]. In general, the current computer architectures are increasingly imbalanced where the latency gap between multi-core CPUs and mechanical hard disks is growing every year which makes the challenges of data-intensive computing harder to overcome [1]. A core challenge in the context of Cloud computing is the management of very large volumes of data. This is completely independent of the type of resource which is shared in the Cloud – databases are either directly visible and accessible to customers as part of the infrastructure/platform, or are hidden behind service interfaces. this means that data need to be partitioned and replicated across different data centers. In Web technology have made it easy for any user to provide and consume content of any form. For example, building a personal Web page (e.g.Google Sites), starting a blog (e.g. WordPress, Blogger, LiveJournal) and making both searchable for the public have now become a commodity. One of the main goals of the next wave is to facilitate the job of implementing every application as a distributed, scalable and widely-accessible service on the Web. It has been recently reported that the famous social network . Big players such as Amazon, Google, have begun to establish new data centers for hosting Cloud computing applications in various locations to provide redundancy and ensure reliability.

1) Infrastructure as a Service (IaaS): Network bandwidth, storage, and related tools necessary to build an application environment.

2) Platform as a Service (PaaS): Load-balancing and scale-out of the platform are done by the service provider and the developer can concentrate on the main functionalities of his application.

3) Software as a Service (SaaS): Manually download, install, configure to use the software applications on their own computing environments.

In the Virtual Machine (VM) that virtualizes physical CPUs (PCPUs) to virtual CPUs (VCPUs), on which guest operating systems run. In a virtual machine system with multi-core processors, usually a VM with multiple VCPUs is treated as a virtual symmetric multiprocessing system. In non-virtualization scenario, all VCPUs in a virtual system are usually not online all the time, and might not be online at the same time when a certain scheduling strategy is running. This is because the number of VCPUs of all VMs is usually larger than the number of PCPUs, and these VCPUs have to share the limited number of PCPUs in turn. As a result, a fair share of CPU should be guaranteed by

Manuscript received May 5, 2014
Manuscript revised May 20, 2014

the Virtual Meachine(VM).VM can be classified into concurrent workloads and high-throughput workloads.
Typically if a VM has to run a mix of several computation-intensive applications, as a simple and efficient method, the first-in-first-out (FIFO) strategy could be adopted. This proportional sharing scheduling method can deliver near-native performance for this kind of workload, and is beneficial to simplify the implementation of the CPU scheduling in the VM.

Virtual Machine: Virtualized server is commonly called a virtual machine (VM). VM allow applications from the underlying hardware and other VMs. The customization of the platform to suit the needs of the end-user. Providers can expose applications running within VMs, or provide access to VMs themselves as a service thereby allowing consumers to install their own applications. Therefore, virtualization forms the foundation of cloud computing, as it provides the capability of pooling computing resources from clusters of servers and dynamically assigning or reassigning virtual resources to applications on-demand. Figure1 illustrates a sample exploitation of Virtual Meachine technology in the cloud computing environments. The VMs are virtual network routers in real machines of the cloud and virtual links are

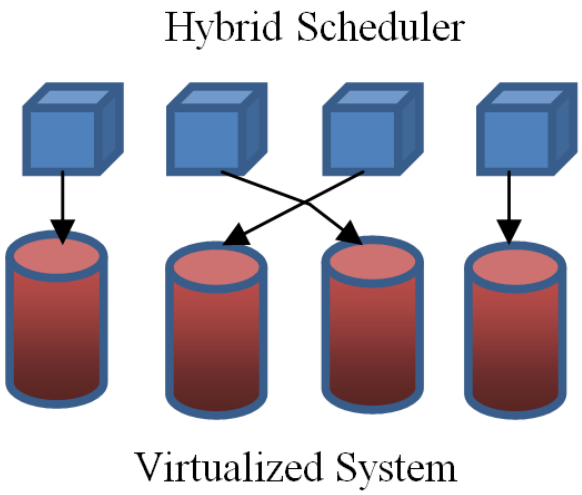

Fig.1:Virtual meachine

Grid Computing: Cloud computing is similar to Grid computing in that it also employs distributed resources to achieve application level objective. In cloud computing takes Pivot level of virtualization technologies at multiple levels to realize resource sharing and dynamic resource provisioning.

Cloud Computing: In the business concept using software as a service, users are provided access to application software and databases. The cloud providers manage the infrastructure and platforms on which the applications run. End users access cloud-based applications through a web browser or a Tablets or mobile app while the business software and user's data are stored on servers at a remote location. Cloud computing allows enterprises to get their applications up and run faster, with improved manageability and less maintenance, and enables IT to more rapidly adjust resources to meet fluctuating and unpredictable business service

1.1 Contribution

In the propose function a hybrid CPU scheduling framework for the VM. In a virtualized system, VM are clustered into two categories based on the characteristics of applications, namely high-throughput VM and concurrent VM. The hybrid scheduling framework performs a hybrid scheduling operation for the different kinds of VM in a virtualized system. In the Time Allocation schedule strategy for high-throughput VM and use the scheduling strategy for concurrent VM.
In the hybrid scheduler framework a comprehensive experimental study to evaluate its performance. Compared with the default credit scheduler in the Hybrid Scheduler performs much better. Moreover, compared with the CPS scheduling strategy, the performance of concurrent workloads in a virtualized system has been further promoted by the PCPS scheduling strategy.

1.2 Scheduling Issue.

Amount of the CPU time obtained by a VM should be controlled. There are two issues with the CPU scheduling in the VM, on which multiple virtual SMP systems run simultaneously. One issue is how to deal with the synchronization issue such as the lock-holder preemption [2] for a virtual SMP system. When multiple VM share a single physical system, it is not a good practice to allow a VM with a heavy workload to appropriate the CPU resource of other VM without any limitation. The scheduler in the VM should keep fairness in resource sharing among VM.

1.3 Scenario

In this non-coscheduling strategy, Fig. 2 (a) depicts a possible scheduling sequence of the above described multithreaded program It is observed that the multithreaded program only completes two steps in the cycle of 10 slots, with 4 slots of CPU time wasted due to synchronization.
There are two kinds of scheduling strategies. One is to asynchronously assign each CPU to a PCPU in order to maximize the throughput, while guaranteeing CPU fairness according to the weights. With this non-coscheduling strategy, Fig.2(a) depicts a possible scheduling sequence of the above described multithreaded program. It is observed

that the multithreaded program only completes two steps in the cycle of 10 slots, with 4 slots of CPU time wasted due to synchronization. Alternative strategy is a co-scheduling strategy, which synchronously assigns each CPU to a PCPU in order to avoid the negative influence of virtualization on synchronization, while guaranteeing CPU fairness according to the weights. Fig.2(b) shows a possible scheduling sequence of the multithreaded program generated by this co-scheduling strategy, and it is observed that the multithreaded program completes three steps within the cycle of 10 slots.

The co-scheduling strategy outperforms the non-coscheduling strategy for concurrent work. the coscheduling strategy considers the correlation of VCPUs, on which a concurrent workload runs, and provides an effective solution to the synchronization problem for concurrent workloads, whereas concurrency is not considered by the non-coscheduling strategy.

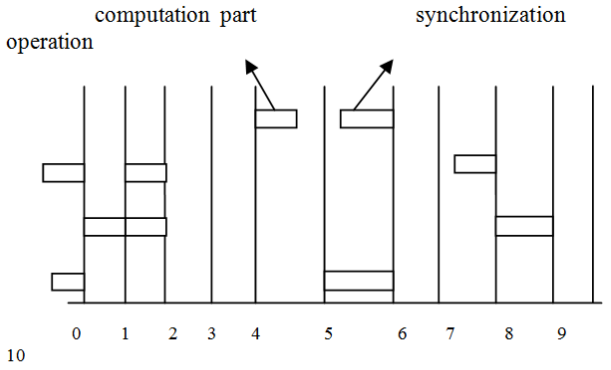

Fig 2:(a)

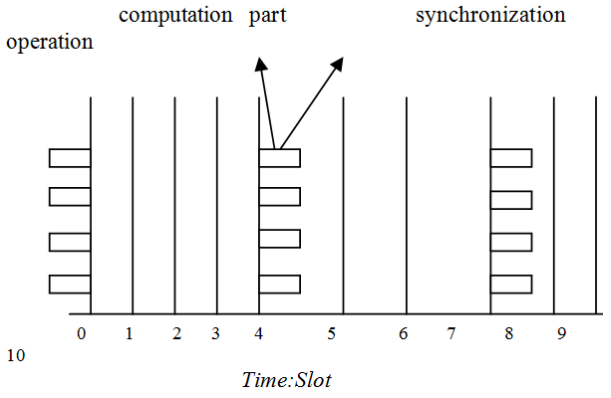

Fig 2:(b)

2. SCHEDULING ANALYSIS

The virtualized system, and then propose the coproportional share (CPS) scheduling strategy that guarantees the fairness of resource allocation and meanwhile coschedules CPUs for concurrent workloads. It extend the CPS scheduling strategy to the phase-co proportional share (PCPS) scheduling strategy to handle the cases when the master task has a longer phase compared to other tasks in concurrent applications.

2.1 Modeling

Scheduling model. In the virtual machine system,the assignment of a concurrent job in a VM includes:
the assignment of jobs on CPUs by the guest operating system of CPUs on PCPUs by the VM. It includes a group of homogenous PCPUs, denoted by P = {P0, P1, ..., P|P|−1}, and the number of PCPUs is |P|. VMs running on a physical computer are denoted by V = {V0, V1, ..., V|V |−1}, and |V | denotes the number of VMs in the system. The weight proportion of VM Vi is denoted by ω(Vi), which represents the proportion of CPU time consumed by the VM, with _i
ω(Vi) = 1.
To avoid expensive switching cost of CPUs mapping on PCPUs, the following relation exists, $|C(Vi)| \leq |P|$. For the same reason, we assume that $|J| \leq |C(Vi)|$. The scheduling problem of a concurrent job J in VM Vi is formalized by $\chi = (\tau, \pi)$, where $\tau : J \rightarrow \{0, 1, 2, ...,\infty\}$, and $\pi : J \rightarrow P$. τ maps phases of tasks to the set of time slots, and π maps phases of tasks to the set of PCPUs in the system. Each task of a concurrent job runs in sequence and they synchronize with each other at the end of phases in an interval, we have $\tau (Jm\ k) + 1 \leq \tau (Jn\ l)$ if $m < n$. It make span of job J execute on VM Vi by the schedule χ is *make span*$\chi(J) = \max k \{\tau (J|Jk|\ k) + 1\}$.

3. SCHEDULING FRAMEWORK

Hybrid scheduling framework to deal with the CPU scheduling problem when multiple VMs with a variety of workloads co-exist in a virtualized system.

3.1 VM classification

In high-throughput application, the scheduling goal is to maximize its throughput.web server application as an example, the performance goal is to maximize the number of access requests that the server can handle, assuming that threads of requests are independent from each other[3]. VMs in a virtualized system can be divided into two kinds, high-throughput VMs and concurrent VMs. When a VM is set as a concurrent VM, CPUs in the VM will be scheduled by the CPS or PCPS strategy. CPUs are scheduled by the PS strategy.

A.Amazon: Dynamo Millions customers at peak times using tens of thousands servers located in many data centers around the world. In this environment, there are strict operational requirements in terms of performance,

reliability and efficiency, and to be able to support continuous growth the platform needs to be highly scalable. Reliability is one of the most important requirements because even the slightest outage has significant financial consequences and impacts customer trust. Amazon has developed a number of storage technologies such as Dynamo System.

B. S3 / SimpleDB / RDS: S3 is an infinite store for objects of variable sizes. An object is simply a byte container which is identified by a URI. Clients can read and update S3 objects remotely using a simple web services interfaces.

4. REPLICATION:PROGRAM MODELS

In the PCPS strategy for a concurrent VM, and then the performance of VM is retested with the PCPS strategy when benchmarks are Parallel Scheduling and Load Scheduling, of sequential parts and parallel parts, and the sequential parts are executed in the master task.

A. MapReduce:

Two main function: 1) *Scaling up*: aims at getting a bigger machine. 2) *Scaling out*: aims at partitioning data across more machines. The scaling up option has the main drawback that large, machines are often very expensive and eventually a physical limit is reached where a more powerful machine cannot be purchased at any cost. For the type of large web application that most people aspire to build, it is either impossible or not cost-effective to run off of one machine. Alternatively, it is both extensible and economical to scale out by adding storage space.

Map Reduce is a programming model that enables easy development of scalable parallel applications to process vast amounts of data on large clusters of commodity machines. [4]. The model is mainly designed to achieve high performance on large clusters of commodity PCs. The computation takes a set of input key/value pairs and produces a set of output key/value pairs.

The Map Reduce abstraction is inspired by the *Map* and *Reduce* functions which are commonly used in the functional languages such as *Lisp(*Locator/Identifier Separation Protocol.). The Map function takes an input pair and produces a set of intermediate key/value pairs. The Map Reduce framework Fig 3 Reduce function receives an intermediate key *K* with its set of values and merges them together. Typically just zero or one output value is produced per Reduce invocation. advantage of this models is that it allows large computations to be easily parallelized and re-execution to be used as the primary mechanism for fault tolerance. Example Map[4] Reduce program expressed in pseudo-code for counting the number of occurrences of each word in a collection of documents. In this example, the map function emits each word plus an associated mark of occurrences while the reduce function sums together all marks emitted for a particular word.

Performance in SMP: MapReduce[7] framework is designed to run on large clusters of commodity hardware. This hardware is managed and powered by open-source operating systems and utilities so that the cost is low

Example:

Mapreduce Program:

map(string name, int i) value)	**reduce(string key,iterator**
name;//document name	**key://a word**
i:// document value	**value:// a list of count**
for each word i is value	**for each v in values**
Emitintermediate(i,"1")	**result+=integer.parseInt(v)**
	Emit(AsString(result));

B. Implementation

One implementation may be suitable for a small shared-memory machine, another for a large NUMA multiprocessor, and yet another for an even larger collection of networked machines. Implementation of Map Reduce that is targeted to the computing environment in wide use at Individual machines typically have 1gigabit/second of network bandwidth,Storage is provided by inexpensive IDE disks attached directly to individual machines. In a distributed file system developed in-house is used to manage the data stored on these disks.

C. Data Structures

Map task and reduce task, it stores the state (idle, in-progress, or completed) and the identity of the worker machine (for non idle tasks). The master is the conduit through which the location of intermediate file regions is propagated from map tasks to reduce tasks. Updates to this location and size information are received as map tasks are completed. The information is pushed incrementally to workers that have in-progress reduce tasks.

.

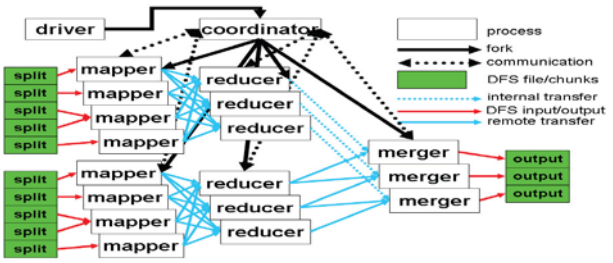

Fig 3: MapReduce Framework

5. Execution in multiple VMs

There are 6 VMs (*V*0,*V*1, ...,*V*5) various combinations of these VMs, in which multiple VMs run on the system simultaneously.*V*0 is also configured as in, workload on it. The configuration of VMs. VMs are configured to use the work-conserving mode[5]. VM is eligible to receive extra CPU time if other VMs are blocked Workloads and performance metrics.

Vm	No.of CPU	Memory	Weight
V0	6	1024	256
.			
.			
V5	8	1024	256

Table: VM Execution

The number of repetitions of each workload is set to be large enough that all other workloads are still running when each workload finishes its first 10 rounds. The result is the average value of the run times of the first 10 rounds for each workload.

5.1 VM time allocation:

VMs time allocation to the weight proportion assigned to them. PE is adopted for CPU time allocation, and the PE value of each CPU is reset to the weight proportion in a certain interval.

VM time allocation Algorithm

Algorithm:
Step 1: calculate Vm where
$Vmtotal \leftarrow |V| \times Vmunit \times L;$

Step 2: for each VM Vi do
Start
$Vminc \leftarrow Vmtotal \times \omega(Vi);$
End

Step 3 if Vi is concurrent to VM then
$W(vi0) \leftarrow W(vi0) + Winc \times 1 + uf(vi0) |C(Vi)| + uf(vi0)$;
for each CPU vij (j > 0) in VM Vi do
Start
$W(vij) \leftarrow W(vij) + Winc\ |C(Vi)| + uf(vi0)$;
End
End
End

Scheduling Algorithm

Algorithm:
for each CPU *Ck* do
Start
if $W(Vm\ (Ck)) > 0$ then
if $VCT(Vm\ (Ck)) = CON$ and *Vm* (C*k*) is a master CPU
then
Lookup VM as *Vm* (C*k*),
C*k* sends Inter-Processor Interrupts
PV (C*k*) is scheduled as
uf(*Vm*(*Ck*))
else if traverse its run queue until it finds the first
vij , where $V\ CT(vij) = HIT$ then
VCPU *vij* is scheduled to CPU C*k*;
end
else
Start
Lookup $vi*j*$ at the head of the run
queue of C*k*_ (k _ _= k), where $PE(vi*j*) = \max$
k_ $PE(Vm\ (Ck\_))$, and $runq(Ck) \cap C(Vi*) = \varphi$ if
$V\ T(Vi*) = CON$;
Migrate $vi*j*$ to *runq*(C*k*) and schedule it to CPU C*k*;
End
End

6. CONCLUSION:

The continuous increase of computational power has produced an overwhelming flow of data. the size of the *digital universe* was about 0.18 zettabyte in the *fourth paradigm* now a days the usage of data is 1.8 zettabyte (a zettabyte is one billion terabytes).,the data growing should be increasing in forth coming generation. the amount of data that is being produced and the capacity of traditional systems to store, analyze and make the best use of this data Cloud computing has gained much momentum in recent years due to its economic advantages. The process of hybrid scheduler for CPU management in the VM. To propose CPS and PCPS strategies for concurrent workloads. The diversity of VM in the cloud platform, a VM can be set as the concurrent type The priority of workloads are concurrent applications, and then the PCPS strategy is adopted to mitigate the negative influence of virtualization on synchronization. it is set as the high-throughput type as the default, and the PS strategy is adopted. Function schedule the hybrid CPU management is Efficient to improve the performance of concurrent applications, maintaining the performance of high-throughput Functions, which run simultaneously on VMs in the cloud platform.

References

[1] T. Hey, S. Tansly, and K. Tolle, editors. The Fourth Paradigm: Data- Intensive Scientific Discovery. October 2009.

[2] V. Uhlig, J. LeVasseur, E. Skoglund, and U. Dannowski. Towards scalable multiprocessor virtual machines. In Proceedings of the 3rd Virtual Machine Research & Technology Symposium, San Jose, CA, 2004.

[3] D. Kossmann, T. Kraska, and S. Loesing. An evaluation of alternative architectures for transaction processing in the cloud. In SIGMOD, 2010.
[4] J.Dean and S.Ghemawat.Mapreduce: simplified data processing on large clusters. ACM, 51(1):107–113, 2008.
[5] L. Cherkasova, D. Gupta, and A. Vahdat. Comparison of the three CPU schedulers in Xen. ACM SIGMETRICS Performance Evaluation Review, 35(2):42–51, 2007.
[6] Siani Pearson,Yun Shen and Miranda Mowbray,"A Privacy For Cloud Computing", HP Labs, Long Down Avenue, Stoke Gifford,Bristol BS34 8QZ, UK.
[7] C.Wang, J.Wang, X. Lin, W.Wang, H.Wang, H.Li, W. Tian, J.Xu, and R. Li. Map Dup Reducer: detecting near duplicates over massive datasets. In SIGMOD, 2010.
[8] S. Gilbert and N. Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. SIGACT News, 33(2):51–59, 2002
[9] Y. Zhang, H. Franke, J. Moreira, and A. Sivasubramaniam. Improving parallel job scheduling by combining gang scheduling and backfilling techniques. In Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS), pages 133–142, 2000.
[10] CreditScheduler.http://wiki.xensource.com/xenwiki/creditscheduler.

K.AISHWARYA received degree M.E from Anna University Chennai in 2011 and joined as an Assistant Professor in various engineering colleges in Tamil Nadu affiliated to Anna University and has four year teaching experience. She has a membership in computer society of India. She has published international journal. Her current primary areas of research is Network Security

R.SATHYAN doing M.E in deparment of Computer Science and Engineering at Arulmigu meenakshi Amman college of engineering Affiliated to Anna University. His research interest in Web Programming languages, Distributed System, Grid Computing and Cloud Computing.