"Dynamic resource allocation in Cloud Computing"

Rupali Shelke [1]
Lecturer in Computer Department,
1Marathwada Mitra Mandal's Polytechnic, Pune,
Maharashtra, India

Rakesh Rajani [2]
Professor, Department of Computer Engineering,
2 Allard College of Engg. And Mgmt., Pune,
Maharashtra, India

Abstract- Cloud computing has become a new age technology that has got huge potentials in enterprises and markets. Clouds can make it possible to access applications and associated data from anywhere. Today Cloud computing is on demand as it offers dynamic flexible resource allocation, for reliable and guaranteed services in pay-as-you-use manner, to Cloud service users. So there must be a provision that all resources are made available to requesting users in efficient manner to satisfy customer's need. Cloud Computing is gaining acceptance in many IT organizations, as an elastic, flexible and variable-cost way to deploy their service platforms using outsourced resources. Major Cloud computing companies have started to integrate frameworks for parallel data processing in their product portfolio, making it easy for customers to access these services and to deploy their programs. In Cloud computing multiple cloud users can request number of cloud services simultaneously. So there must be a provision that all resources are made available to requesting user in efficient manner to satisfy their need. In this paper we are reviewing such various policies for resource allocation in cloud computing based on Service-Level- Agreement (SLA), centralized decision and distributed multiple criteria decision Parallel data processing framework reduces time and cost in processing the substantial amount of users' data.

Keywords- Cloud Computing; Cloud Services; Resource Allocation, high-throughput computing, Map Reduce

I. INTRODUCTION

A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers. Cloud computing emerges as a new computing paradigm which aims to provide reliable, customized and QoS (Quality of Service) guaranteed computing dynamic environments for end-users. Cloud computing is the delivery of computing as a service rather than a product, whereby shared resources, software and information are provided to users over the network.

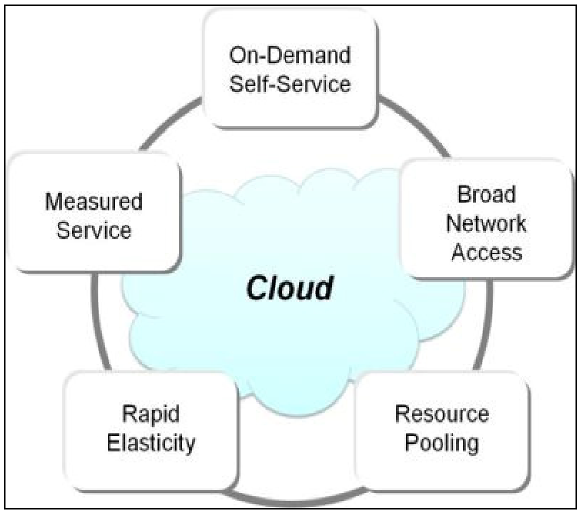

Figure 1.0 Cloud Customer Requirement

Cloud computing providers deliver application via the Internet, which are accessed from web browser, while the business software and data are stored on servers at a remote location. Cloud providers are able to attain the agreed SLA, by scheduling resources in efficient manner and by deploying application on proper VM as per the SLA objective and at the same time performance of the applications must be optimized. Compared to previous paradigms, cloud computing focuses on treating computational resources as measurable and billable utilities. From the clients' point of view, cloud computing provides an abstraction of the underlying hardware architecture. This abstraction saves them the costs of design, setup and maintenance of a data center to host their Application Environments (AE). Whereas for cloud providers, the arrangement yields an opportunity to profit by hosting many AEs. This economy of scale provides benefits to both parties, but leaves the providers in a position where they must have an efficient and cost effective data center . The main goal is to decrease the overloads of the main cloud and increase the performance of the cloud. In recent years ad-hoc parallel data processing has emerged to be one of the killer applications for Infrastructure-as-a-Service (IaaS) clouds. Major Cloud computing companies have started to integrate frameworks for parallel data processing in their product portfolio, making it easy for customers to access these services and to deploy their programs. However, the processing frameworks which are currently used have been designed for static, homogeneous cluster setups and disregard the particular nature of a cloud.

II. RELATED WORK

Recently there was lot of research went on parallel data processing and its implications and possibilities. Many systems came into existence for processing MTC applications where parallel processing of data is essential. All such systems have a common goal such as fault tolerance and parallel execution of tasks and they are being used in different fields. Open source version of MapReduce i.e. Hadoop or the MapReduce were designed to run jobs in parallel in cost effective manner using commodity servers. For simplicity an example framework is MapReduce. Once job is given to it, it automatically takes care of dividing the given job into tasks and spreading them across the available servers. There are two programs involved namely Map and Reduce for specific functionality. There are many other programs that coordinate with the jobs of MapReduce nature. MapReduce is designed to run data analysis jobs on a large amount of data, which is expected to be stored across a large set of share-nothing commodity servers. MapReduce is highlighted by its simplicity: Once a user has fit his program into the required map and reduce pattern, the execution framework takes care of splitting the job into subtasks, distributing and executing them. A single MapReduce job always consists of a distinct map and reduce program. MapReduce has been clearly designed for large static clusters. Although it can deal with sporadic node failures, the available compute resources are essentially considered to be a fixed set of homogeneous machines.

III. CHALLENGES AND OPPORTUNITIES

Today's processing frameworks typically assume the resources they manage consist of a static set of homogeneous compute nodes. Although designed to deal with individual nodes failures, they consider the number of available machines to be constant, especially when scheduling the processing job's execution. While IaaS clouds can certainly be used to create such cluster-like setups, much of their flexibility remains unused. To integrate the cloud computing task such as portfolio access these services and to deploy their programs for efficient parallel processing. Each vertex in the graph represents process flow edges define the communication between these tasks aslo decided to use Directed Acyclic Graph is relevant to Nephele. The user must write the program code for external task must be assigned to a vertex and must be connected by edges to define the communication paths of the job. In this paper we have discussed the challenges and opportunities for efficient parallel data processing in cloud environments and presented Nephele, the first data processing framework to exploit the dynamic resource provisioning offered by today's IaaS clouds. We have described Nephele's basic architecture and presented a performance comparison to the well-established data processing framework Hadoop.

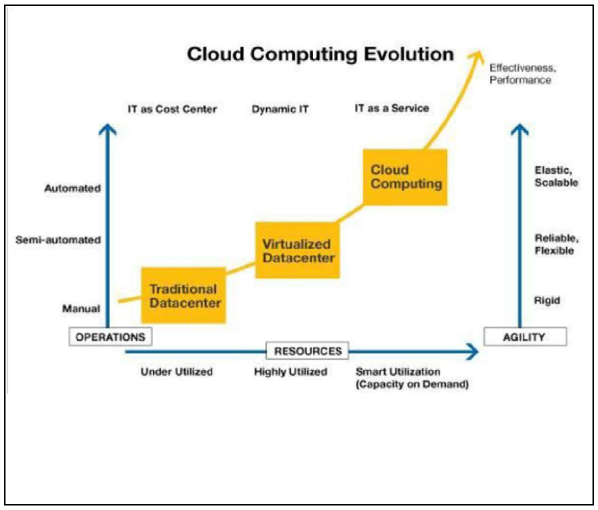

Figure 2.0 Cloud Scenarios and Demand

IV. RESOURCE ALLOCATION AND ITS SIGNIFICANCE

In cloud computing, Resource Allocation (RA) is the process of assigning available resources to the needed cloud applications over the internet. Resource allocation starves services if the allocation is not managed precisely. Resource provisioning solves that problem by allowing the service providers to manage the resources for each individual module. Resource Allocation Strategy (RAS) is all about integrating cloud provider activities for utilizing and allocating scarce resources within the limit of cloud environment so as to meet the needs of the cloud application. It requires the type and amount of resources needed by each application in order to complete a user job. The order and time of allocation of resources are also an input for an optimal RAS. An optimal RAS should avoid the following criteria as follows:

- Resource Contention - Resource contention arises when two applications try to access the same resource at the same time.
- Scarcity of Resource - Scarcity of resource arises when there are limited resources and the demand for resources is high.
- Resource Fragmentation - Resource fragmentation arises when the resources are isolated. There would be enough resources but cannot allocate it to the needed application due to fragmentation into small entities.

- Over Provisioning - Over provisioning arises when the application gets surplus resources than the demanded one.
- Under Provisioning - Under provisioning of resources occurs when the application is assigned with fewer numbers of resources than it demanded.

V. PROBLEM DEFINITION

The efficient parallel data processing is achieved by using Nephele's framework by dynamic resource allocation offered by today's IaaS clouds for both, task scheduling and execution. Particular tasks of a processing job can be assigned to different types of virtual machines which are automatically instantiated and terminated during the job execution. Based on this, we perform extended evaluations of MapReduce-inspired processing jobs on an IaaS cloud system. The processing framework then takes care of distributing the program among the available nodes and executes each instance of the program on the appropriate fragment of data. Most notably, Nephele is the first data processing framework to include the possibility of dynamically allocating/ de-allocating different compute resources from a cloud in its scheduling and during job execution.

VI. DYNAMIC RESOURCE ALLOCATION FOR PARALLEL DATA PROCESSING

Dynamic Resource Allocation for Efficient Parallel data processing introduces a new processing framework explicitly designed for cloud environments called Nephele. Most notably, Nephele is the first data processing framework to include the possibility of dynamically allocating/de- allocating different compute resources from a cloud in its scheduling and during job execution. Particular tasks of a processing job can be assigned to different types of virtual machines which are automatically instantiated and terminated during the job execution. Nephele's architecture follows a classic master-worker pattern as illustrated in Figure 3.0. Before submitting a Nephele compute job, a user must start a VM in the cloud which runs the so called Job Manager (JM).The Job Manager receives the client's jobs, is responsible for scheduling them, and coordinates their execution. It is capable of communicating with the interface the cloud operator provides to control the instantiation of VMs. We call this interface the Cloud Controller. By means of the Cloud Controller the Job Manager can allocate or de-allocate VMs according to the current job execution phase. The actual execution of tasks which a Nephele job consists of is carried out by a set of instances. Each instance runs a so called Task Manager (TM). A Task Manager receives one or more tasks from the Job Manager at a time, executes them, and after that informs the Job Manager about their completion or possible errors.

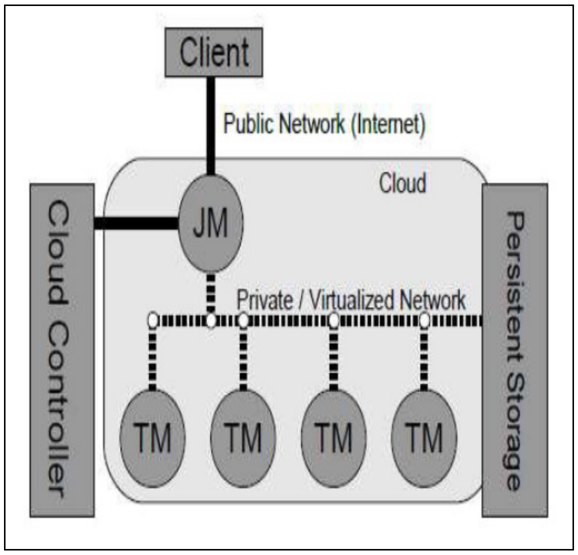

Figure 3.0 Nephele Architecture

VII. ARCHITECTURE

A. Query Processing

Processing is based on implementation of the theorem uses (network-based) search operations as off the shelf building blocks. Thus, the NAP query evaluation methodology is readily deployable on existing systems, and can be easily adapted to different network storage schemes. In this case, the queries are evaluated in a batch. we propose the network-based anonymization and processing (NAP) framework, the first system for K-anonymous query processing in road networks. NAP relies on a global user ordering and bucketization that satisfies reciprocity and guarantees K-anonymity. We identify the ordering characteristics that affect subsequent processing, and qualitatively compare alternatives. Then, we propose query evaluation techniques that exploit these characteristics.

B. Parallelization

One fundamental idea to refine the scheduling strategy for recurring jobs is to use feedback data. We developed a profiling subsystem for Nephele+ which can continuously monitor running tasks and the underlying instances. Based on the Java Management Extensions (JMX) the profiling subsystem is, among other things, capable of breaking down what percentage of its processing time a task thread actually spends processing user code and what percentage of time it has to wait for data. With the collected data Nephele+ is able to detect both computational as well as I/O bottlenecks. While computational bottlenecks suggest a higher degree of parallelization for the affected tasks, I/O bottlenecks provide hints to switch to faster channel types.(like in-memory channels) and reconsider the instance assignment. Since Nephele+ calculates a cryptographic signature for each task, recurring tasks can be identified and the previously recorded feedback data can be exploited.

C. Advantages

Major advantage of resource allocation is that user neither has to install software nor hardware to access the applications, to develop the application and to host the application over the internet. Also there is no limitation of place and medium. We can reach our applications and data anywhere in the world, on any system. Cloud providers can share their resources over the internet during resource scarcity.

VIII. CONCLUSION

The author highlights use cases which are currently poorly supported by existing parallel data processing frameworks and explains how a tighter integration between the processing framework and the underlying cloud system can help to lower the monetary processing cost for the cloud customer. With a framework like Nephele at hand, there are a variety of open research issues, which we plan to address for future work. In particular, we are interested in improving Nephele's ability to adapt to resource overload or underutilization during the job execution automatically. Our current profiling approach builds a valuable basis for this; however, at the moment the system still requires a reasonable amount of user annotations.

IX. REFERENCES

1. Atsuo Inomata, TaikiMorikawa, Minoru Ikebe, Sk.Md. MizanurRahman: Proposal and Evaluation of Dynamin Resource Allocation Method Based on the Load Of VMs on IaaS(IEEE,2010),978-1-4244-8704-2/11.
2. "Adaptive Management of Virtualized Resources in Cloud Computing Using Feedback Control," in First International Conference on Information Science and Engineering, April 2010, pp. 99-102.
3. Buyya, R., Yeo, C. S., & Venugopal, S. (2008). Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In HPCC '08: Proceedings of the 2008
4. 10th IEEE International Conference on High Performance Computing and Communications (pp. 5{13). Washington, DC, USA: IEEE Computer Society.
5. Jiyani et al.: Adaptive resource allocation for preemptable jobs in cloud systems (IEEE, 2010), pp.31-36.
6. J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," Proc. Sixth Conf. Symp. Opearting Systems Design and Implementation (OSDI '04), p. 10, 2004.
7. R. Chaiken, B. Jenkins, P.-A. Larson, B. Ramsey, D. Shakib, S. Weaver, and J. Zhou. SCOPE: Easy and Efficient Parallel Processing of Massive Data Sets. Proc. VLDB Endow., 1(2):1265–1276, 2008.
8. H. chih Yang, A. Dasdan, R.-L. Hsiao, and D. S. Parker. Map-Reduce-Merge: Simplified Relational Data Processing on Large Clusters. In SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, pages 1029–1040,New York, NY, USA, 2007. ACM.
9. Daniel Warneke and Odej Kao, Exploiting dynamic resource allocation for efficient parallel data processing in the cloud, IEEE Transactions On Parallel And Distributed Systems, 2011.
10. C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins. Pig Latin: A Not-So-Foreign Language for Data Processing. In SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pages 1099–1110, New York, NY, USA, 2008. ACM.
11. E. Deelman, G. Singh, M.-H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. B. Berriman, J. Good, Laity, J. C. Jacob, and D. S. Katz. Pegasus: A Framework for Mapping Complex Scientific Workflows onto Distributed Systems. Sci. Program., 13(3):219–237, 2005.