Available online at www.elixirpublishers.com (Elixir International Journal)

Computer Science and Engineering

Elixir Comp. Sci. & Engg. 53 (2012) 12035-12037

ISSN: 2229-712X

Computing over a multi cloud for MTC applications

Challa Vanitha Reddy and Battula Sudheer Kumar
C.V.S.R College Of Engineering, JNTU.

ARTICLE INFO

Article history:
Received: 15 April 2012;
Received in revised form:
7 December 2012;
Accepted: 12 December 2012;

Keywords
Cloud Computing,
MTC applications,
HTC.

ABSTRACT
IT organizations can now outsource computer hardware by leasing CPU time through cloud computing services. The problem here is the effectiveness is becoming less due to burden on single cloud while working with MTC applications .This paper deals with defining the feasible solutions with MTC applications using the programming models for computing over a Multi clouds instead of single cloud for effectiveness.

Introduction

This decade is marked by a paradigm shift of the industrial information technology towards a subscription based or pay-per-use service business model known as cloud computing. This paradigm provides users with a long list of advantages, such as provision computing capabilities; broad, heterogeneous network access; resource pooling and rapid elasticity with measured services [1].

The single cloud will not be more effective for MTC paradigm (MAny-Task Computing (MTC) paradigm [2] embraces different types of high-performance applications involving many different tasks, and requiring large number of computational resources over short periods of time. These tasks can be of very different nature, with sizes from small to large, loosely coupled or tightly coupled, or compute-intensive or data-intensive). As the single cloud will not be efficient for MTC paradigm we can also go with Computing Clusters.

Computing clusters have been one of the most popular platforms for solving MTC problems, specially in the case of loosely coupled tasks (e.g. high-throughput computing applications). However, building and managing physical clusters exhibits several drawbacks: 1) major investments in hardware, specialized installations (cooling, power, etc.), and qualified personal; 2) long periods of cluster under-utilization; 3) cluster overloading and insufficient computational resources during peak demand periods. To over come the above draw backs in cluster computing we go for computing clusters over a multi cloud.

The simultaneous use of different cloud providers to deploy a computing cluster spanning different clouds can provide several benefits:

• High-availability and fault tolerance, the cluster worker nodes can be spread on different cloud sites, so in case of cloud downtime or failure, the cluster operation will not be disrupted. Furthermore, in this situation, we can dynamically deploy new cluster nodes in a different cloud to avoid the degradation of the cluster performance.

• Infrastructure cost reduction, since different cloud providers can follow different pricing strategies, and even variable pricing models (based on the level of demand of a particular resource type, daytime versus night-time, weekdays versus weekends, spot prices, and so forth), the different cluster nodes can change dynamically their locations, from one cloud provider to another one, in order to reduce the overall infrastructure cost [3].

The rest of this paper explains about the Background, Related work, MTC application through programming model and Conclusion.

Back Ground

Many-task computing aims to bridge the gap between two computing paradigms, high throughput computing and high performance computing. Many task computing differs from high throughput computing in the emphasis of using large number of computing resources over short periods of time to accomplish many computational tasks (i.e. including both dependent and independent tasks), where primary metrics are measured in seconds (e.g. FLOPS, tasks/sec, MB/s I/O rates), as opposed to operations (e.g. jobs) per month. Many task computing denotes high-performance computations comprising multiple distinct activities, coupled via file system operations [4]. Tasks may be small or large, uniprocessor or multiprocessor, computeintensive or data-intensive. The set of tasks may be static or dynamic, homogeneous or heterogeneous, loosely coupled or tightly coupled. The aggregate number of tasks, quantity of computing, and volumes of data may be extremely large. Many task computing includes loosely coupled applications that are generally communication-intensive but not naturally expressed using standard message passing interface commonly found in high performance computing, drawing attention to the many computations that are heterogeneous but not “happily” parallel.

Related Work

We have found many applications that are a better fit for MTC than HTC or HPC. Their characteristics include having a large number of small parallel jobs, a common pattern observed in many scientific applications [5]. They also use files (instead messages, as in MPI) for intra-processor communication, which tends to make these applications data intensive. While we can push hundreds or even thousands of such small jobs via GRAM

Tele:
E-mail addresses: vanithareddy52@gmail.com

to a traditional local resource manager (e.g. Condor [6]), the achieved utilization of a modest to large resource set will be poor due to high queuing and dispatching overheads, which ultimately results in low job throughput. A common technique to amortize the costs of the local resource management is to “cluster” multiple jobs into a single larger job. Although this lowers the per job overhead, it is best suited when the set of jobs to be executed are homogenous in execution times, or accurate execution time information is available prior to job execution; with heterogeneous job execution times, it is hard to maintain good load balancing of the underlying resource, causing low resource utilization. We claim that “clustering” jobs is not enough, and that the middleware that manages jobs must be streamlined and made as light-weight as possible to allow applications with heterogonous execution times to execute without “clustering” with high efficiency.

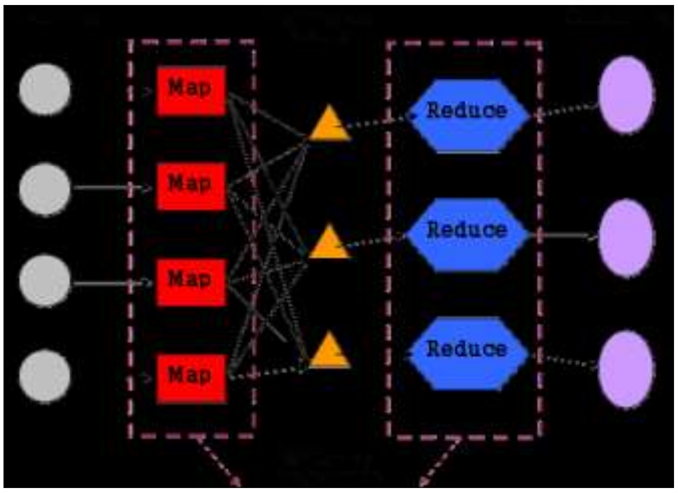
In addition to streamlined task dispatching, scalable data management techniques are also required in order to support MTC applications [7]. MTC applications are often data and/or meta-data intensive, as each job requires at least one input file and one output file, and can sometimes involve many files per job. These data management techniques need to make good utilization of the full network bandwidth of large scale systems, which is a function of the number of nodes and networking technology employed, as opposed to the relatively small number of storage servers that are behind a parallel file system or GridFTP server. We have identified various applications (as detailed below) from many disciplines that demonstrate characteristics of MTC applications. These applications cover a wide range of domains, from astronomy, physics, astrophysics, pharmaceuticals, bioinformatics, biometrics, neuroscience, medical imaging, chemistry, climate modeling, economics, and data analytics. They often involve many tasks, ranging from tens of thousands to billions of tasks, and have a large variance of task execution times ranging from hundreds of milliseconds to hours. Furthermore, each task is involved in multiple reads and writes to and from files, which can range in size from kilobytes to gigabytes. These characteristics made traditional resource management techniques found in HTC inefficient; also, although some of these applications could be coded as HPC applications, due to the wide variance of the arrival rate of tasks from many users, an HPC implementation would also yield poor utilization. Furthermore, the data intensive nature of these applications can quickly saturate parallel file systems at even modest computing scales.

MTC Application through Programming Model

MTC applications through programming models and tools give the feasible and efficient solution for MTC application.

MTC through Programming model (Map Reduce)

MapReduce [8] is one of the most popular programming models designed to support the development of compute and data-intensive applications which have high storage and processing demands which were initially met by large scale computer systems. MapReduce was initially created by Google for simplifying the development of large scale web search applications in data centers and has been proposed to form the basis of a ‘Data center computer’. MapReduce model is suitable for web search services, scientific research projects referred to as e-Science and data mining application amongst others.



MTC application is executed in a parallel manner through two phases in Map reduce model. In the first phase, MTC application are submitted to map functions and those operations are known as map operations. all map operations can be executed independently with each other. In the second phase, each reduce operation may depend on the outputs generated by any number of map operations. However, similar to map operations, all reduce operations can be exe-cuted independently.

From the perspective of dataflow, MapReduce execution for MTC applications consists of m independent map tasks and r independent reduce tasks, each of which may be dependent on m map tasks. Generally the intermediate results are partitioned into r pieces for r reduce tasks[9].

The MapReduce runtime system schedules map and reduce tasks to distributed resources. It manages many technical problems: parallelization, concurrency control, network com-munication, and fault tolerance. Furthermore, it performs several optimizations to de-crease overhead involved in scheduling, network communication and intermediate group-ing of results. Thus the MTC application runs very efficiently through map reduce model.

Conclusion

In this paper we have shown how MTC applications in multi cloud will be more efficient and feasible than in single cloud and also the applications with support of Programming model give the feasible and efficient solution. It takes even very less time to compute them.

Reference:

[1] P. Mell, T. Grance, “Draft NIST working definition of cloud computing”, Referenced on June. 3rd, 2009, Online at http://csrc.nist.gov/groups/SNS/cloud-computing/index.html, 2009.

[2] I. Raicu, I. Foster, Y. Zhao, Many-Task Computing for Grids and Supercomputers, Workshop on Many-Task Computing on Grids and Supercomputers, 2008, pp. 1–11

[3] E. Walker, The Real Cost of a CPU Hour. Computer 42(4): 35–41, 2009.

[4] J. Ousterhout, “Scripting: Higher Level Programming for the 21st Century”, IEEE Computer, March 1998

[5] Y. Zhao, M. Hategan, B. Clifford, I. Foster, G. von Laszewski, I. Raicu, T. Stef-Praun, M. Wilde. “Swift: Fast, Reliable, Loosely Coupled Parallel Computation”, IEEE Workshop on Scientific Workflows 2007

[6] J. Frey, T. Tannenbaum, I. Foster, M. Frey, S. Tuecke. “Condor-G: A Computation Management Agent for Multi-Institutional Grids”, Cluster Computing, 2002.

[7] E.V. Hensbergen, Ron Minnich. “System Support for Many Task Computing”, IEEE Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS08) 2008
[8] J. Dean, S. Ghemawat. “MapReduce: Simplified data processing on large clusters.” In OSDI, 2004
[9] M. Cole, Parallel Programming with List Homomorphisms, Parallel Processing Letters 5 (1995) 191–203.

Sudheer Kumar Battula received the Bachelor’s degree from Aurora’s Scientific and Technological Institute affliated to JNTU Hyderabad University in 2009. I’m currently pursuing my Masters from Anurag Group of Institutions affliated to JNTU Hyderabad University in Computer Science & Engineering department.

Vanitha Reddy Challa received the Bachelor’s degree from Aurora’s Scientific and Technological Institute, affliated to JNTU Hyderabad University in 2009. I’m currently pursuing my Masters from *Anurag* Group of Institutions affliated to JNTU Hyderabad University in Computer Science & Engineering department