ISSN 2349-4506
Impact Factor: 2.265

Global Journal of Engineering Science and Research Management

EXPLOITING DYNAMIC RESOURCE ALLOCATION

M. Gowri Shankar*, Mrs G.S.Geethamani
*M.Phil, Research Scholars, Department of Computer Science, Hindusthan College of Arts and Science, Coimbatore.
Assistant Professor Department of Information Technology and Computer Technology, Hindusthan College of Arts and Science Coimbatore.

KEYWORDS: Single Processing, Multiprocessing, Cloud Computing, Neples Algorithm.

ABSTRACT

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. Consequently, the allocated compute resources may be inadequate for big parts of the submitted job and unnecessarily increase processing time and cost. We discuss the opportunities and challenges for efficient parallel data processing in clouds and present our research project Nephele. Nephele is the first data processing framework to explicitly exploit the 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 new framework, we perform extended evaluations of Map Reduce-inspired processing jobs on an IaaS cloud system and compare the results to the popular data processing framework Hadoop.

INTRODUCTION

Today a growing number of companies have to process huge amounts of data in a cost-efficient manner. More companies have popularized an architectural paradigm based on a large number of commodity servers. Problems like processing crawled documents or regenerating a web index are split into several independent subtasks, distributed among the available nodes, and computed in parallel.

In order to simplify the development of distributed applications on top of such architectures, many of these companies have also built customized data processing frameworks. Examples are Google's MapReduce, Microsoft's Dryad , or Yahoo!'s Map-Reduce-Merge .They can be classified by terms like high throughput computing (HTC) or many-task computing (MTC), depending on the amount of data and the number of tasks involved in the computation . Although these systems differ in design, their programming models share similar objectives, namely hiding the hassle of parallel programming, fault tolerance, and execution optimizations from the developer. Developers can typically continue to write sequential programs. 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

OPTIMIZING MULTIWAY JOINS IN A MAP-REDUCE ENVIRONMENT

Implementations of map-reduce are being used to perform many operations on very large data. We examine strategies for joining several relations in the map-reduce environment. Our new approach begins by identifying the "map-key," the set of attributes that identify the Reduce process to which a Map process must send a particular tuple. Each attribute of the map-key gets a "share," which is the number of buckets into which its values are hashed, to form a component of the identifier of a Reduce process. Relations have their tuples replicated in limited fashion, the degree of replication depending on the shares for those map-key attributes that are missing from their schema. We study the problem of optimizing the shares, given a fixed number of Reduce processes. An algorithm for detecting and fixing problems where a variable is mistakenly included in the map-key is given. Then, we consider two important special cases: chain joins and star joins.

In each case, we are able to determine the map-key and determine the shares that yield the least replication. While the method we propose is not always superior to the conventional way of using map-reduce to implement joins, there are some important cases involving large-scale data where our method wins, including: 1) analytic queries in which a very large fact table is joined with smaller dimension tables, and 2) queries involving paths through graphs with high out-degree, such as the Web or a social network.

ISSN 2349-4506
Impact Factor: 2.265

Global Journal of Engineering Science and Research Management

MAP-REDUCE-MERGE: SIMPLIFIED RELATIONAL DATA PROCESSING ON LARGE CLUSTERS

We design and implement Mars, a Map Reduce framework, on graphics processors (GPUs). Map Reduce is a distributed programming framework originally proposed by Google for the ease of development of web search applications on a large number of commodity CPUs. Compared with CPUs, GPUs have an order of magnitude higher computation power and memory bandwidth, but are harder to program since their architectures are designed as a special-purpose co-processor and their programming interfaces are typically for graphics applications. As the first attempt to harness GPU's power for Map Reduce, we developed Mars on an NVIDIA G80 GPU, which contains over one hundred processors, and evaluated it in comparison with Phoenix, the state-of-the-art Map Reduce framework on multi-core CPUs. Mars hides the programming complexity of the GPU behind the simple and familiar

IMPLEMENTATION

Cloud computing has emerged as a promising approach to rent a large IT infrastructure on a short-term pay-per-usage basis. Operators of so-called Infrastructure-as-a-Service (IaaS) clouds, like Amazon EC2, let their customers allocate, access, and control a set of virtual machines (VMs) which run inside their data centers and only charge them for the period of time the machines are allocated. In the proposed system Nephele, a new processing framework explicitly designed for cloud environments.

Nephele is the first data processing framework to include the possibility of dynamically allocating or deallocating different compute resources from a cloud in its scheduling and during job execution. 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.

NEPHELE ALGORITHM

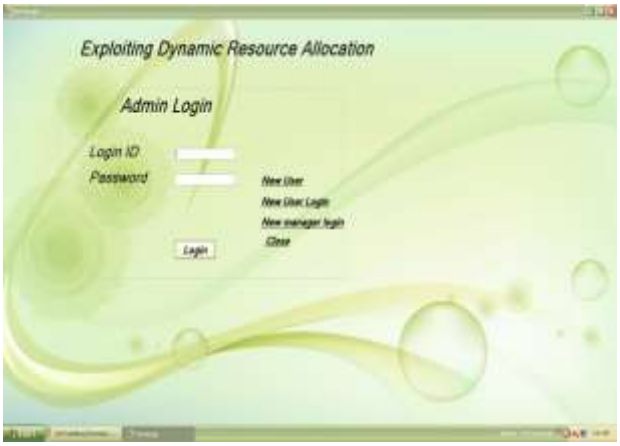
A data processing framework with support for dynamic allocation and de-allocation of different computational resources in the cloud.

Compute resources available in a cloud environment are highly dynamic and possibly heterogeneous. In addition, the network topology is hidden so scheduling optimizations based on knowledge of the distance to a particular rack or server are impossible. The classic genotyping approach has been based on phylogenetic analysis, starting with a multiple sequence alignment. Genotypes are then established by expert examination of phylogenetic trees. However, such methods are suboptimal for a rapidly growing dataset, because they require significant human effort, and because they increase in computational complexity quickly with the number of sequences. This project uses a method for genotyping that does not depend on multiple sequence alignment. It uses the complete composition vector algorithm to represent each sequence in the dataset as a vector derived from its constituent k-mers, and affinity propagation clustering to group the sequences into genotypes based on a distance measure over the vectors. Our methods produce results that correlate well with expert-defined clades or genotypes, at a fraction of the computational cost of traditional phylogenetic methods.

Increasingly, genotyping is coming into use for disease surveillance activities, as well as for microbial forensics. The classic genotyping approach has been based on phylogenetic analysis, starting with a multiple sequence alignment. Genotypes are then established by expert examination of phylogenetic trees. However, these traditional single-processor methods are suboptimal for rapidly growing sequence datasets being generated by next-generation DNA sequencing machines, because they increase in computational complexity quickly with the number of sequences.

ISSN 2349-4506
Impact Factor: 2.265

Global Journal of Engineering Science and Research Management



MODULES DEPLOYMENT

7.1 SERVER TYPE SUPPORT

The scheduler is given a list of available server types and their cost per time unit. Each task can be executed on its own server type.. An Execution Instance has an ID and an server type representing the hardware characteristics.

7.2 TRAFFIC IDENTIFIER

All edges of an Execution Graph are replaced by a channel before processing can begin. There are three channel types: A network channel is based on a TCP socket connection. Two subtasks connected via a network channel can be executed on different instances. Since they must be executed at the same time, they are required to run in the same Execution Stage.

7.35STORAGE MANAGEMENT SYSTEM

After having received a valid Job Graph from the user, Nephele's Job Manager transforms it into a so-called Execution Graph. An Execution Graph is Nephele's primary PARALLEL AND data structure for scheduling and monitoring the execution of a Nephele job.

ISSN 2349-4506
Impact Factor: 2.265

Global Journal of Engineering Science and Research Management

Each vertex of the Job Graph is transformed into one Execution Vertex. The default channel types are network channels. Each Execution Vertex is by default assigned to its own Execution Instance unless the user's annotations or other scheduling restrictions (e.g. the usage of in-memory channels) prohibit it. One fundamental idea to refine the scheduling strategy for recurring jobs is to use feedback data. With the collected data Nephele is able to detect both computational as well as I/O bottlenecks. we only use the profiling data to detect these bottlenecks.

CONCLUSION

The challenges and opportunities for efficient parallel data processing in cloud environments are discussed and presented Nephele, the first data processing framework to exploit the dynamic resource provisioning offered by today's IaaS clouds. Nephele's basic architecture and presented a performance comparison to the well-established data processing framework Hadoop.

The performance evaluation gives a first impression on how the ability to assign specific virtual machine types to specific tasks of a processing job, as well as the possibility to automatically Allocate/deallocate virtual machines in the course of a job execution, can help to improve the overall resource utilization and, consequently, reduce the processing cost. 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.

ISSN 2349-4506
Impact Factor: 2.265

Global Journal of Engineering Science and Research Management

Our current profiling approach builds a valuable basis for this, however, at the moment the system still requires a reasonable amount of user annotations.

REFERENCES

1. Amazon Web Services LLC. Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/, 2009.
2. Amazon Web Services LLC. Amazon Elastic MapReduce.http://aws.amazon.com/elasticmapreduce/, 2009.
3. AmazonWeb Services LLC. Amazon Simple Storage Service. http: //aws.amazon.com/s3/, 2009.
4. D. Battr´e, S. Ewen, F. Hueske, O. Kao, V. Markl, and D. Warneke. Nephele/PACTs: A Programming Model and Execution Framework for Web-Scale Analytical Processing. In SoCC ’10: Proceedings of the ACM Symposium on Cloud Computing 2010, pages 119– 130, New York, NY, USA, 2010. ACM.
5. 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.