Exploiting Dynamic Resource Allocation for Query Processing in the Cloud Computing

M.S.B.Pridviraju , K.Rekha Devi

Department of CSE , Kakatiya Institute of Technology & Science, Warangal Dist-506002,India.

Abstract—In recent years unplanned 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 collections, making it easy for customers to access these services and to deploy their programs. 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 not sufficient for big parts of the submitted job and unnecessarily increase processing time and cost. In this paper 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.

Keywords- Many-Task Computing, queryprocessing, Cloud Computing

I. INTRODUCTION

Cloud computing is the technology used to access remotely stored data through the internet. It protects the data from the disasters like earthquakes, tsunami, cyclones, fire etc. Cloud computing protects the data by using emails, personal records, documents, etc. By storing the useful data into the cloud, the owners are free from the burden of maintenance. In this, owners can share their data with the large number of users when the users request for the data. The users might wants to retrieve only specific data files in which they are interested. Today a growing number of companies have to process huge amounts of data in a cost-efficient manner. Classic representatives for these companies are operators of Internet search engines, like Google, Yahoo, or Microsoft. The vast amount of data they have to deal with every day has made traditional database solutions are expensive .Instead, these companies have popularized an architectural paradigm based on a large number of commodity servers. Problems like processing 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 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 differ in design, their programming models share similar objectives, namely 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. For companies that only have to process large amount of data occasionally running their own data center is obviously not an option. Instead, Cloud computing has emerged as a promising approach to rent a large IT infrastructure on a short-term pay-per-usage basis. Opera-tors of so-called Infrastructure-as-a-Service (IaaS) clouds, access, and control a set of virtual machines (VMs) which run inside their data centres and only charge them for the period of time the machines are allocated .As a result, rented resources may be inadequate for big parts of the processing job, which may lower the overall processing performance and increase the cost. In this paper we want to discuss the particular challenges and opportunities for efficient parallel data pro- cessing in clouds and present Nephele, a new processing framework explicitly designed for cloud environments. Most notably, Nephele is the first data processing frame-work to include the possibility of dynamically allocating/deal locating different compute resources from a cloud in its scheduling and during job execution.

2. CHALLENGES AND OPPORTUNITIES

Current data processing frameworks like Google's MapReduce or Microsoft's Dryad engine have been de-signed for cluster environments. This is reflected in a number of assumptions they make which are not necessarily valid in cloud environments. In this section we discuss how abandoning these assumptions raises new opportunities but also challenges for efficient parallel data processing in clouds.

Opportunities

Today's processing frameworks typically assume the resources they manage consist of a static set of homogeneous compute nodes. New VMs can be allocated at any time through a well defined interface and become available in a matter of seconds. Machines which are no longer used can be terminated instantly and the cloud customer will be charged for them no more.

Challenges

The cloud's virtualized nature helps to enable promising new use cases for efficient parallel data processing. How ever, it also imposes new challenges compared to classic

cluster setups. The major challenge we see is the cloud's opaqueness with prospect to exploiting data locality: In a cluster the compute nodes are typically interconnected through a physical high-performance network.

3. DESIGN

Based on the challenges and opportunities outlined in the previous section we have designed Nephele, a New data processing framework for cloud environments. Nephele takes up many ideas of previous processing frameworks but refines them to better match the dynamic and opaque nature of a cloud.

3.1. ARCHITECTURE

Nephele's architecture follows a classic master-worker pattern as illustrated in Fig.

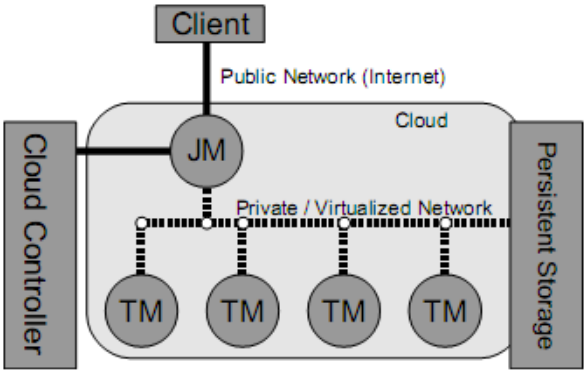


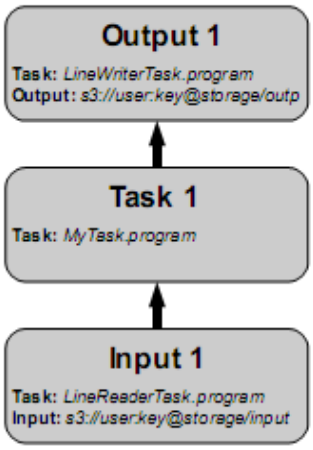
Figure 1 : Queryprocessing

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. Cloud Controller. By means of the Cloud Controller the Job Manager can allocate or deal locate 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.

Unless a job is submitted to the Job Manager. We expect the set of instances to be empty. When the respective instances must be allocated/deal located to ensure a continuous but cost-efficient processing. The newly allocated instances boot up with a previ- ously compiled VM image. The image is configured to automatically start a Task Manager and register it with the Job Manager. Once all the necessary Task Managers have successfully contacted the Job Manager, it triggers the execution of the scheduled job. Initially, the VM images used to boot up the Task Managers are blank and do not contain any of the data the Nephele job is supposed to operate on if they are connected by a private or virtual network.

Job description

Similar to Microsoft's Dryad jobs in Nephele are Expressed as a directed acyclic graph (DAG). Each vertex in the graph represents a task of the overall processing job, the graph's edges define the communication flow between these tasks. We also decided to use DAGs to describe processing jobs for two major reasons: The first reason is that DAGs allow tasks to have multiple input and multiple output edges. This tremendously simplifies the implementation of classic data



ig. 2. An example of a Job Graph in Nephele

I.RELATED WORK

A growing number of companies have to process huge amounts of data in a cost-efficient manner. Classic representatives for these companies are operators of Internet search engines. The vast amount of data they have to deal with every day has made traditional database solutions prohibitively

Expensive .Instead, these 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.

II. PROPOSED SYSTEM

In recent years a variety of systems to facilitate MTC has been developed. Although these systems typically share common goals (e.g. to hide issues of parallelism or fault tolerance), they aim at different fields of application. 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.

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 Map Reduce job always consists of a distinct map and reduce program.

NETWORK MODULE:

Server - Client computing or networking is a distributed application architecture that partitions tasks or workloads between service providers (servers) and service requesters, called clients. A client also shares any of its resources. Initiate communication sessions with servers which await incoming requests.

LBS SERVICES

Linking a position to an individual is possible by various means, such as publicly available information city maps.

Even though a user may create a fake ID to access the service, her location alone may disclose her actual identity. When a user u wishes to pose a query, she sends her location to a trusted server.

SYSTEM MODEL

Each vertex of the job system model graph is transformed in to one execution vertex If constructing an execution graph from a user submitted job graph may leave different degrees of freedom to nephele

SCHEDULED TASK:

Recently, considerable research interest has focused on preventing identity inference in location-based services. This offers privacy protection in the sense that the actual user position u cannot be distinguished from others in the ASR (anonym zing spatial region) even when malicious LS is advanced enough to possess all user locations. This spatial K-anonymity model is most widely used in location privacy research/applications, even though alternative models are emerging.

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. NAP achieves low computational and communication costs, and quick responses overall. It is readily deployable, requiring only basic network operations.

Job Scheduling and Execution

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 data structure for scheduling and monitoring the execution of a Nephele job. Unlike the abstract Job Graph, the Execution Graph contains all the concrete information required to schedule and execute the received job on the cloud. It explicitly models task parallelization and the mapping of tasks to instances. Depending on the level of annotations the user has provided with his Job Graph, Nephele may have different degrees of freedom in constructing the Execution Graph.

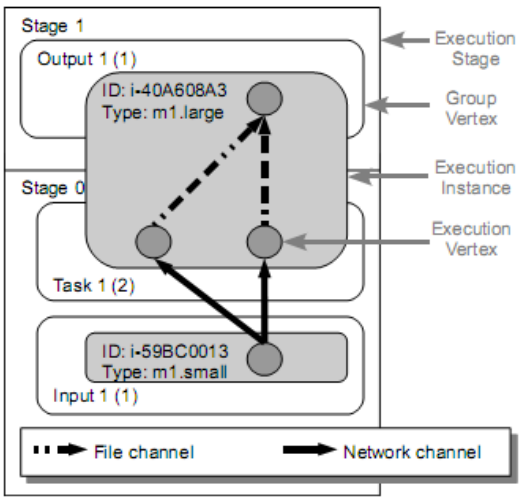


Fig. 3. An Execution Graph created from the original Job Graph

While the abstract graph describes the job execution on a In contrast to the Job Graph, an Execution Graph is no longer a pure DAG. Instead, its structure resembles a graph with two different levels of details, an abstract and a concrete level task level (without parallelization) and the scheduling of instance allocation/deal location, the concrete, more fine-grained graph defines the mapping of subtasks to instances and the communication channels between them.

Parallelization and Scheduling Strategies:

If constructing an Execution Graph from a user's submitted Job Graph may leave different degrees of freedom to Nephele. The user provides any job annotation which contains more specific instructions we currently pursue simple default strategy: 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.

IV RESULTS

The concept of this paper is implemented and different results are shown below. The proposed paper is implemented in .net technology on a System Pentium IV 2.4 GHz. Hard Disk40 GB. Floppy Drive1.44 Mb Monitor 15 VGA Colour. Mouse Logitech. Ram 512 MB. The propose paper's concepts shows 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.

V.CONCLUSION

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. 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/deal locate 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. Our current profiling approach builds a valuable basis for this, however, at the moment the system still requires a reasonable amount of user annotations. In general, we think our work represents an important contribution to the growing field of Cloud computing services and points out exciting new opportunities in the field of parallel data processing.

REFERENCES

Daniel Warneke is a research assistant at the Berlin University of Technology, Germany. Hereceived his Diploma and BS degrees in computer science from the University of Paderborn, in 2008 and 2006, respectively. Daniel's research interests centre around massively parallel; fault-tolerant data processing frame works on Infrastructure-as-a-Service platforms. Currently, he is working in the DFG-funded research project Stratosphere.

Odej Kao is full professor at the Berlin University of Technology and director of the IT center tub IT. He received his PhD and his habilitation from the Clausthal University of Technology. There- after, he moved to the University of Paderborn as an associated professor for operating and distributed systems. His research areas include Grid Computing, service level agreements and operation of complex IT systems. Odej is a member of many program committees and has published more than 190 papers. Good Teachers are worth more than thousand books, we have them in Our Department

[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.

[6] 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.

[7] M. Coates, R. Castro, R. Nowak, M. Gadhiok, R. King, and Y. Tsang. Maximum Likelihood Network Topology Identification from Edge-Based Unicast Measurements. SIGMETRICS Perform. Eval. Rev., 30(1):11–20, 2002.

[8] R. Davoli. VDE: Virtual Distributed Ethernet. Testbeds and Research Infrastructures for the Development of Networks & Communities, International Conference on, 0:213–220, 2005.

[9] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.

[10] E. Deelman, G. Singh, M.-H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. B. Berriman, J. Good, A. 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.

[11] T. Dornemann, E. Juhnke, and B. Freisleben. On-Demand Resource Provisioning for BPEL Workflows Using Amazon's Elastic Compute Cloud. In CCGRID '09: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, pages 140–147, Washington, DC, USA, 2009. IEEE Computer Society.