Chapter 10
Ad-Hoc Parallel Data Processing on Pay-As-You-Go Clouds with Nephele

Daniel Warneke
Technische Universität Berlin, Germany

ABSTRACT
In recent years, so-called Infrastructure as a Service (IaaS) clouds have become increasingly popular as a flexible and inexpensive platform for ad-hoc parallel data processing. Major players in the cloud computing space like Amazon EC2 have already recognized this trend and started to create special offers which bundle their compute platform with existing software frameworks for these kinds of applications. However, the data processing frameworks which are currently used in these offers have been designed for static, homogeneous cluster systems and do not support the new features which distinguish the cloud platform. This chapter examines the characteristics of IaaS clouds with special regard to massively-parallel data processing. 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. As a proof of concept, the author presents the parallel data processing framework Nephele, and compares its cost efficiency against the one of the well-known software Hadoop.

INTRODUCTION
During the last decade, the number of companies and institutions which have to process huge amounts of data has increased rapidly. While operators of Internet search engines like Google, Yahoo!, or Microsoft are still prominent examples for these kinds of companies, today we can also constitute a growing demand for large-scale data analysis from scientific institutions (Gray et al., 2005) and companies whose business focus has traditionally been outside of the information management space (Gonzalez, Han, Li, & Klabjan, 2006).
Processing data at a scale of several tera- or even petabytes either goes far beyond the scalability of traditional parallel database systems or entails licensing costs which render such solutions prohibitively expensive for most institutions (Chaiken et al, 2008). Instead, recent price developments for commodity hardware and multi-core CPUs have made architectures consisting of large sets of inexpensive commodity servers the preferred choice for large-scale data processing in recent years.

While these new distributed architectures offer tremendous amounts of compute power at an unprecedented price point, they also complicate the development of applications. Developers are suddenly confronted with the difficulties of parallel programming in order to write code which scales well to hundreds or even thousands of CPUs. Moreover, since individual nodes from the large set of compute resources are likely to fail, the programs must also be written in a way to tolerate a certain degree of hardware outages.

In order to simplify the development of distributed applications for these new compute platforms, several operators of Internet search engines, which have also pioneered the new architectural paradigm, have introduced customized data processing frameworks. Examples are Google’s MapReduce (Dean & Ghemawat, 2008), Microsoft’s Dryad (Isard, Budiu, Yu, Birrell, & Fetterly, 2007), or Yahoo!’s Map-Reduce-Merge (Yang, Dasdan, Hsiao, & Parker, 2007). Although these systems differ in design, their programming models have common goals, namely hiding the hassle of parallel programming, fault tolerance, and execution optimizations from the developer. While developers can typically continue to write sequential programs, the parallel data processing framework takes care of deploying these programs among the commodity servers and executing the parallel instances of them on the appropriate parts of the input data.

Despite the price decline for commodity hardware, a reasonable application of such parallel data processing frameworks has so far been limited to those companies which have specialized in the field of large-scale data analysis and therefore operate their own data centers. However, through eliminating the need for large upfront capital expenses, cloud computing now also enables companies and institutions which only have to process large amounts of data occasionally to gain access to a highly scalable pool of compute resources on a short-term pay-per-usage basis.

Operators of so-called Infrastructure as a Service (IaaS) clouds, like Amazon Web Services (Amazon Web Services, Inc. [AWS], 2011), Rackspace (Rackspace US, Inc. [RS], 2011), or GoGrid (GoGrid, LLC [GG], 2011) let their customers rent compute and storage resources hosted within their data centers. The size of these cloud data centers typically adds up to thousands or ten thousands of servers, creating the impression of virtually unlimited resources to the customer. In order to simplify their deployment, the compute resources are typically offered in form of virtual machines which are available with different hardware properties (such as compute power, amount of main memory, disk space, etc...) and at different costs. The usage of the leased resources is charged on a short-term basis (for example, compute resources by the hour and storage by the day or month), thereby rewarding conservation by releasing machines and storage when they are no longer needed (Armbrust et al., 2010).

Since the virtual machine abstraction of IaaS clouds fits the architectural paradigm assumed by the parallel data processing frameworks described above, first major cloud computing companies have started to combine existing processing frameworks with their IaaS platform and offer these bundles as a separate product. A prominent example is Amazon Elastic MapReduce (EMR) (AWS, 2011b) which runs the parallel data processing framework Hadoop (The Apache Software Foundation [ASF], 2011) on top of Amazon’s Elastic Compute Cloud (EC2) (AWS, 2011c).