Stratus: scientific simulations in the cloud

Pelle Jakovits, Satish Narayana Srirama
Institute of Computer Science, University of Tartu, J. Liivi 2, Tartu, Estonia
{jakovits, srirama}@ut.ee

I. Introduction

Scientific computing is a field of computer science that deals with solving large-scale scientific problems in domains like astrophysics, mechanical engineering and material science by utilizing mathematical modeling and computer simulations. Running large and accurate simulations requires a significant amount of computing resources, often demanding the utilization of supercomputers, computer clusters or grids. Thus, scientific computing has historically been dependent on the advances of High Performance Computing (HPC) and parallel processing. In the past few years, in addition to supercomputers and grids, cloud computing [1] has proved itself as a new way to acquire HPC resources on demand.

However, researchers who require HPC resources for their domain specific work are often not intimate with writing, configuring and debugging parallel applications, so only providing the infrastructure is not enough and additional tools, like MapReduce [2], are needed to simplify their work.

We have studied [3] using MapReduce for solving scientific computing problems and found that while it is very good for embarrassingly parallel algorithms, it does not fully support complex iterative algorithms that are very often used in scientific simulations. This motivated us to look for other distributed computing models that would be more suitable for our needs. One such model is Bulk Synchronous Parallel (BSP).

II. Bulk Synchronous Parallel model

Bulk Synchronous Parallel [4] (BSP) is a distributed computing model for iterative algorithms where the whole task is divided into concurrent sub-tasks and the computation consists of a sequence of super-steps. Each super-step can be divided into three sub-steps: local computation, where a user defined function is applied to every sub-task concurrently, synchronization step, where data is communicated between (neighboring) sub-tasks and global barrier, where all sub-tasks wait until every other sub-task has also finished.

A typical BSP application is defined by how the whole computation is divided into concurrent sub-tasks, how the data is synchronized and what function is applied to each sub-task at every super-step. Rest of the tasks needed to perform computations and to manage the parallelism of the whole algorithm should be handled automatically.

While there exist a number of tools that utilize BSP, we have found they are not fully usable for our needs because they are either developed for legacy platforms, are still under heavy development, or are strictly designed for other purposes, like processing graphs. For this reason, we have decided to develop a new framework based on the BSP model, called Stratus.

III. Stratus framework

The goal of the framework is to provide a distributed computing platform for performing large scale scientific computing simulations and experiments on the cloud. The framework consists of a single Cluster Manager (CM) and a number of Task Managers (TM) that are set up as a dynamic computer cluster. On cloud infrastructure, it is set up on a number of temporary cloud instances that are allocated for each new task based on the amount of required computing resources like memory, computing power or even budget. Input and output is stored in cloud volumes, which act as persistent storage for temporary virtual machines.

The job of the Cluster Manager is to start and maintain user requested tasks, allocate computing and storage resources in the form of TM instances and cloud volumes, deallocate these resources when the tasks are finished, and take care of load balancing and fault recovery. The job of the Task Manager is to manage execution of its subtasks, handle communication between local and remote subtasks, and take care of the barrier synchronization for all the local subtasks.

One of the most important aspects, when deploying long running applications in the cloud, is fault recovery. To achieve fault tolerance in Stratus, Task Managers log outbound communication and periodically store the current state of the tasks as snapshots in the connected volumes. In case of a machine or network failure, the Cluster Manager restores the failed subtasks by combining the latest snapshot and the messages that the failed subtasks received since then. This enables the framework to fully restore the state of the calculations before the failure occurred.

Stratus is being developed in the University of Tartu as part of the Scientific Computing in the Cloud (SciCloud) project.

References

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[2] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Commun. ACM*, vol. 51, pp. 107–113, January 2008.

[3] S. N. Srirama, P. Jakovits, and E. Vainikko, "Adapting scientific computing problems to clouds using mapreduce," *Future Generation Computer Systems*, vol. 28, no. 1, pp. 184–192, 2012.

[4] T. Cheatham, A. Fahmy, D. C. Stefanescu, and L. G. Valiant, "Bulk synchronous parallel computing – a paradigm for transportable software," in *In Proc. IEEE 28th Hawaii Int. Conf. on System Science*. Society Press, 1995, pp. 268–275.