Vilnius university
Institute of Data Science and Digital Technologies
L I T H U A N I A

INFORMATICS ENGINEERING (T007)

DISTRIBUTED PUBLIC COMPUTING FOR BIG DATA MINING

Albertas Jurgelevičius

October 2019

Technical Report DMSTI-DS-T007-19-05

VU Institute of Data Science and Digital Technologies, Akademijos str. 4, Vilnius
LT-08412, Lithuania
www.mii.lt

Abstract

Grid computing is a popular but expensive method for companies and organizations to process large amounts of data. Studies show that under low system load it is rational to only use efficient devices for data processing, however, under high system load inefficient devices should be used also. The purpose of this paper is to overview our proposed big data mining platform, which solves the task distribution problem by using hierarchical two-level scheduling and dynamic size task stalling buffer. Results show that our platform reduces total task execution time on average by 17,63%, which also reduces costs and increases the quality of service.

Keywords: Big data mining, public distributed computing, hybrid cloud, scheduling algorithms

Contents

1 Introduction

Grid computing is a popular but expensive method for companies and organizations to process large amounts of data. In some cases, more affordable methods like public cloud computing are used by combining different resource capacity nodes. However, such heterogeneous environment lack reliability. This can be solved by using hybrid cloud computing. The term hybrid cloud computing refers to combining public and private cloud infrastructures for performing computations. In business environment data usually cannot leave the company boundaries and public cloud computing cannot be used. However, in such cases internal infrastructure such as employee computers still may be used. This can solve reliability and cost related issues.

Studies show the dilemma whether or not inefficient devices should be used for computations [4]. If such devices are not used, then the additional processing power is lost. However, if more tasks are assigned to such slow devices, then stuttering may occur. Tasks may be different in size and the more of large tasks are assigned to slow devices, the higher probability of stuttering to occur as usually tasks are not allowed to be cancelled.

We propose to solve this issue by using task stalling buffer. Studies show that under low system load it is rational to only use efficient devices, however, under high load slow devices should be used also. This can be solved by applying appropriate size stalling buffer. Studies show that the size of the stalling buffer should depend on system load, which can be calculated using incoming task rate and idle slow device count. If stalling buffer size is static, then probability of stuttering still remains. This leads to the conclusion that the stalling buffer size should depend on busy slow device count.

The rest of this paper is structured as follow. Section 2 presents our volunteer-based hybrid cloud computing platform. Section 3 overviews the scheduling algorithms. Finally, Sections 4 and 5 concludes this paper by summarizing the results and presenting conclusions.

2 Architecture

2.1 Design philosophy

Hybrid Cloud aims to provide scalable and resilient core for task execution. Our design philosophy has been to push task scheduling to lower level grids and control which grid should receive the task. This method ensures that compute resources from each environment are being used efficiently. First, we check the grid load and then forward the task according to the selected scheduling method, that will be reviewed in Section 3. Next we will overview two lower level grids that we use in our big data mining platform.

2.2 Apache Mesos

As defined in [2], Apache Mesos is a platform for sharing commodity clusters between multiple diverse cluster computing frameworks, such as Hadoop and MPI. One of such frameworks is called Chronos. We use it as part of our platform since it provides native Docker support that we require for virtualization purposes to support different architectures. Results in [2] show that Apache Mesos can achieve near-optimal data locality when sharing the cluster among diverse frameworks, can scale to 50,000 (emulated) nodes, and is resilient to failures.

2.3 BOINC

Berkeley Open Infrastructure for Network Computing (BOINC) is a well-known platform that may be used for both volunteer computing and grid computing. It provides means to take use of compute resources from both home and office computers. BOINC supports running virtualized applications allowing computations on different architectures.

Cloud computing and big data mining using BOINC is well reviewed in [3], showing that public distributed computing approaches may compete with existing cloud computing solutions.

2.4 Hybrid cloud

To reduce the costs and increase the reliability we present a hybrid cloud based on two grids: BOINC and Apache Mesos (reviewed in Sections 2.2 and 2.3). Our proposed architecture (see Figure 1) contains physically distributed (hierarchical) cooperative schedulers, that creates a two-level hierarchy. At the top level there is a master scheduler that distributes tasks between two schedulers at the bottom level. The bottom level consists of two grids, each being managed by a scheduler specifically designed for each environment.

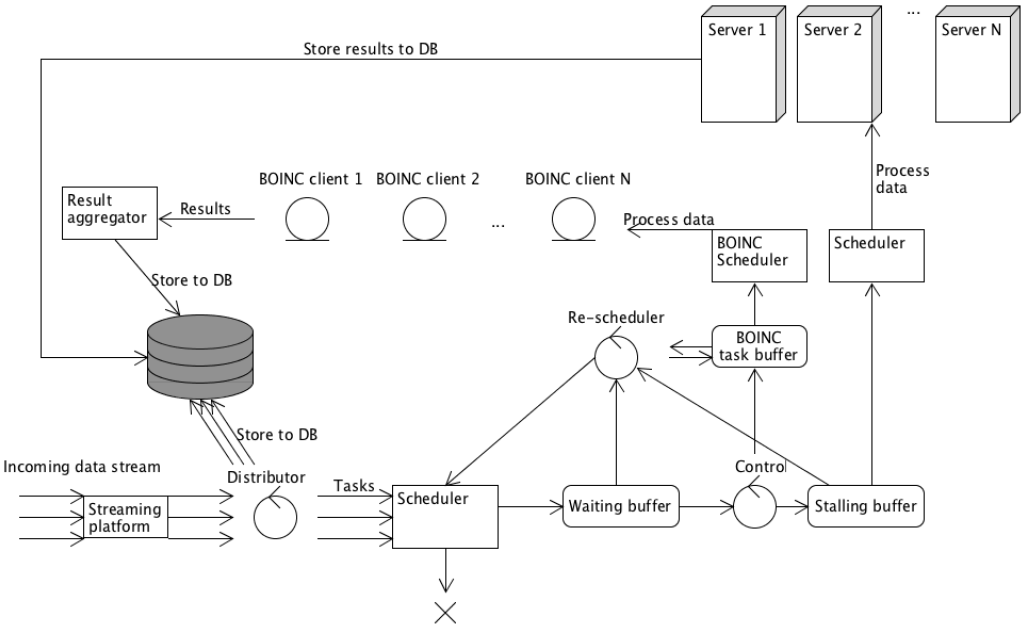

Figure 1. Model

Scheduling algorithms to be used for distributing tasks between the two grids are presented in Section 3.

3 Scheduling algorithms

There are few widely adopted independent job schedulers. Some are used for big data processing by such companies like Facebook and Yahoo [5]:

- *Fair-share - scheduling algorithm used by Facebook [6] that allocates equal share of resources to each job [5];*
- *Capacity scheduler - originally developed by Yahoo [1, 6]. It uses queues instead of pools unlike fair scheduler, and each queue will be assigned to an organization after the resources have been divided among these queues;*

- *LATE scheduler - finds out process running with slow speed in the cluster and creates an equivalent process in the background as a backup [5];*
- *Round-robin - runs all the applications from the first job on the first node, all the applications from the second job on the second node, etc.;*
- *FIFO (First-In-First-Out) – is an opportunistic scheduling method that assigns task to the first compute resource to become available. It is widely used to schedule big data mining tasks and it is the default scheduler for Hadoop [5].*

Round-robin and FIFO are two scheduling algorithms that are capable of distributing incoming dynamic tasks between two grids in real time. However, these methods are inefficient and unreliable in heterogeneous environments, since they do not consider resource performance capabilities. Stalling buffer introduced by [4] is used to improve the opportunistic scheduling method and solve this issue. This scheduling method from [4] was adapted, integrated into our big data mining platform and compared to the standard FIFO scheduling algorithm. Results are presented in Section 4.1.

4 Experiment

Task execution effectiveness was measured by comparing two scheduling algorithms described in Section 3. Experiment was performed by executing distributed π value calculation tasks using Monte Carlo method. Tasks were divided into two categories to test real case scenarios: short running tasks (100000 iterations) and long running tasks (200000 iterations). Furthermore, two different incoming task intensities were used for tests: static (290 seconds between tasks) and dynamic (using Poisson distribution). Two servers running Docker containers were used for the experiment (see Figure 2). Fair compute resource distribution between task execution processes was ensured by adding upper limit for CPU usage per process in virtual environment configurations (Virtualbox and Docker).

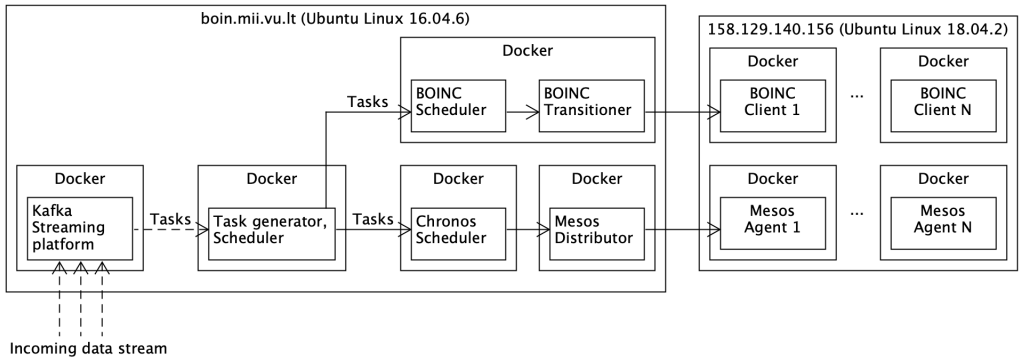

Figure 2. Test environment schema

4.1 Results

The results (see Figure 3) show that total time required to execute 100 tasks is significantly reduced when using opportunistic task scheduling method with dynamic stalling buffer. Total task execution time was reduced on average by 17,63%.

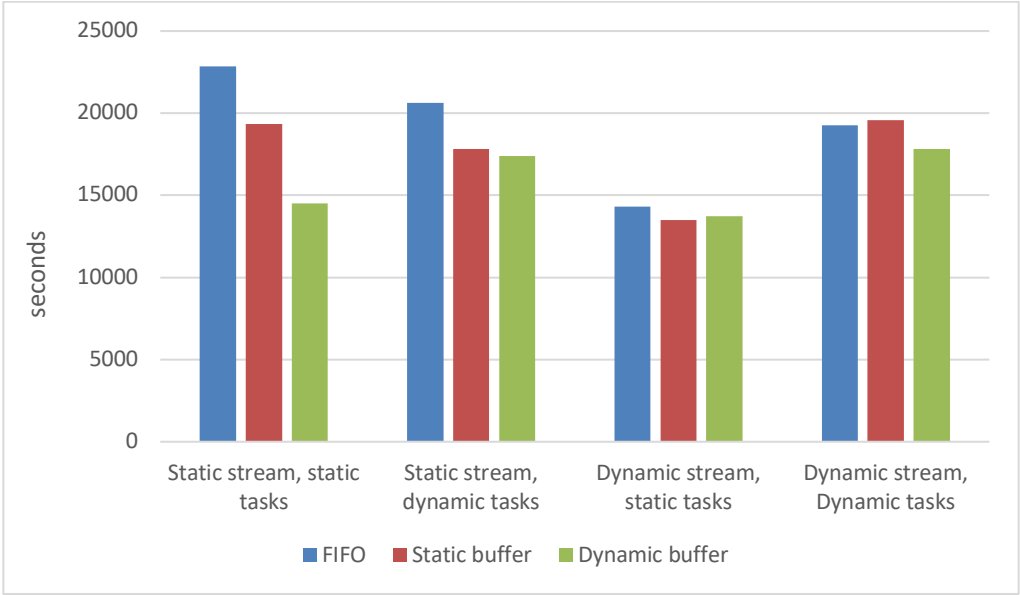

Figure 3. Time required to execute 100 tasks

5 Conclusion

Our proposed big data mining platform is more effective when opportunistic task scheduling method with dynamic stalling buffer is used. Execution time was reduced on average by 17,63%, which also reduces costs and increases task execution reliability. As a result, our proposed big data mining platform also meets the company and organization requirements for such type of platforms.

6 References

[1] Gautam, J.V., Prajapati, H.B., Dabhi, V.K., Chaudhary, S.: A survey on job scheduling algorithms in big data processing. IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT'15), Coimbatore, 2015, 1–11.

[2] Hindman, B., Konwinski, A., Zaharia, M., Ghodsi, A., Joseph, A., Katz, R., Shenker, S., Stoica, I.: Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center. Proceedings of the 8th USENIX conference on Networked systems design and implementation, March 30 - April 01, 2011, 295-308.

[3] Jurgelevičius, A., Sakalauskas, L.: BOINC from View Point of Cloud Computing. COUR Workshop Proceedings, 1973, 2017, 61-66.

[4] Kaklauskas, L., Sakalauskas, L., Denisovas, V.: Stalling for Solving Slow Server Problem. RAIRO-Oper. Res. 53 (4), 2019, 1097-1107. https://doi.org/10.1051/ro/2018056

[5] Usama, M., Liu, M., Chen, M.: Job schedulers for Big data processing in Hadoop environment: testing real-life schedulers using benchmark programs. Digital Communications and Networks 3(4), November, 2017, 260-273. https://doi.org/10.1016/j.dcan.2017.07.008

[6] Yoo, D., Sim, K.M.: A comparative review of job scheduling for MapReduce. Cloud Computing and Intel. Syst. (CCIS), IEEE Int. Conf. on. IEEE, 2011.