International Journal of Advanced Science and Technology
Vol.70 (2014), pp.11-16
http://dx.doi.org/10.14257/ijast.2014.70.02

Review on the Cloud Computing Programming Model

Chao Shen and Weiqin Tong

School of Computer Engineering and Science
Shanghai University, Shanghai 200072, China
shenchao1010@aliyun.com@qq.com, wqtong@shu.edu.cn

Abstract

Cloud computing integrates vast computing and/or storage resources together, which provides services on demand via network. The cloud computing data center is usually composed of thousand of commercial computers, and these computers are connected by network. A cloud computing programming model is urgent to design to help user to use cloud resources without concerning the details of implementation. Nowadays, some cloud computing programming models have been proposed. This paper first analyzes the problems which cloud computing programming model need to solve, and then analyzes the characteristics of the cloud computing programming model. The advantages and disadvantages of each programming model are proposed. Finally, this is paper concludes the current issues and future trends.

Keywords: *Programming Model, Cloud Computing, Parallel Computing*

1. Introduction

Nowadays, the amount of data in many applications has been increased from TB level to PB level, and even higher level. Various fields have accumulated massive data. On the other hand, the price of computer hardware is decline continuously, and the ability of compute and storage is increase continuously. Cloud computing is advent on this background [1].

Cloud computing integrates vast computing and/or storage resources together, which provides services on demand via networks. Developers request resources on demand and pay for it by hours. With the development of network, more and more applications satisfy user needs by network. Application and data are stored on the cloud platform.

The cloud computing programming model is not easy. The cloud computing data center is usually composed of thousand of commercial computers, and these computers are connected by network. So far, it is not well supported by software models, computer languages or tools. The resources utilization is not high. So, there is urgent need to design a cloud computing programming model that is adapted to cloud computing environment. The user can easily programming applications and uses cloud resources efficiently by the programming model.

2. The Main Problems Solved by the Cloud Computing Programming model

Compared with the tradition compute, the cloud computing usually process massive data and it is dynamic and scalable. The cloud computing programming model mainly solves the following problems.

1) Parallel

ISSN: 2005-4238 IJAST

The cloud computing mainly processes massive data. Parallel processing is an efficient method. However, parallel computing is a long-term challenge in computer science. So far, there is still no a unified parallel computing model. Nowadays, the parallel computing is mainly relying on manual design of parallel computing, and there are not proper parallel programming languages and tools.

2) Fault tolerant

The cloud computing center is composed of thousands of cheap commercial computer. The probability of computer fault is high. So, the cloud computing programming model must provide fault tolerant function.

3) Heterogeneous

Some organizations have already a lot of computers before establishing their cloud computing center. The configuration of these computers may be different. On the other hand, if computers of the cloud computing center need to be upgraded, computers can be upgraded gradually, and they need not upgraded at once. This is lead to the heterogeneous of the cloud computing center. So, the cloud computing programming model must solve the heterogeneous.

4) Load balance

Load balance is a very important factor than affect the performance of the cloud computing. If the load is not balance, the cloud resources utilization will be degraded seriously. In order to make full use of the cloud resources, cloud computing programming model must make load balance as far as possible.

3. Cloud Computing Programming Model

Although the cloud computing programming model faces many challenges, some cloud computing programming models have been proposed by many organizations. These cloud computing programming model is mainly includes MapReduce [2] which is based on map and reduce operations, Dryad [3] which implements parallel used by data flow graph, similar SQL languages which use MapReduce or Dryad as execute engine and provide higher level of abstraction, workflow-oriented parallel programming model, and All-pairs [13] that fits the needs of several data intensive scientific applications.

3.1. MapReduce

MapReduce is a programming model and implementation which used to process massive data. Users only need define two functions map and reduce. Map function is used to process a series of key-value pairs, and generate a series of intermediate key-value pairs. These intermediate key-value pairs are written to local disk, and are partitioned into R regions. The compute node which execute reduce function will read intermediate key-value pairs of one region and sort them. These key-value pairs which have the same key are group together. These intermediate key-value pairs are processed by reduce function, and are generated smaller data set.

MapReduce hides the details of parallel, fault tolerant, data allocation, resources scheduling, load balance and etc. Users only need implement two simple functions: map and reduce. The parallel operation can be easily implemented. User can easily use the MapReduce even they have not experience of parallel and distributed programming. MapReduce has good scalability, and it can implement parallel operation on thousands of or even more computers. In addition, it also has some shortcoming. 1) Some applications are hardly expressed by map and reduce. 2) All computes are limited with single input and single output. 3) Codes are not easy maintained and reused.

Hadoop [4] is an open source implementation of the MapReduce. The performance of the Hadoop is mainly relying on its scheduler. The scheduler of the Hadoop compares each task's progress with the average progress. Tasks which are behind the average progress will be backup executed, which can decrease the response time of the system. Hadoop runs well in the homogeneous environment. If it is run in the heterogeneous environment, a lot of backup tasks are launched. These backup tasks occupy a lot of resources, which lead to the performance of the system is decreased seriously.

3.2. Dryad

Dryad [3] is a coarse-grained data parallel computing platform released by Microsoft. A dryad job is a directed acyclic graph where each vertex is a program and edges represent data channels. The vertices provided by the application developer are quite simple and are usually written as sequential programs with no thread creation or locking. The data channel abstraction has several concrete implementations that use shared memory, TCP pipes, or files temporarily persisted in a file system. Dryad run environment automatically map the logical computing graph to physical resources. Vertices are parallel executing on multiple computers or multiple cores of a computer.

Compared with MapReduce, Dryad is more flexible. It allows multiple inputs and multiple outputs. Vertices are usually serial program, and the programming is easier. In addition, it also has some shortcoming. 1) It is not suitable for iterative and nesting program. 2) Some irregular computing can hardly be converted into data flow graph.

3.3. Similar SQL Programming

In the analysis of massive data, the abstract level of MapReduce it too low. The data flow of one input and two stages is very stiff. The joins or multiple stages operation is not easy implemented. In addition, codes are not easy maintained or reused, and even the simple operations are must be rewritten. In order to solve these problems, some programming models provide higher level abstraction based on the MapReduce, and there are also programming models provide higher level abstraction based on the Dryad.

Pig Latin [5] is a programming language which is used to analyze massive data by Yahoo. Pig Latin combines the advantages of the declarative style of SQL and the low-level procedural style of map-reduce. Pig Latin procedure is a sequence of steps where each step species only a single, high-level data transformation. Pig Latin support flexible and nested data models and it also widely support used defined function.

HadoopDB[6] uses PostgreSQL as the database layer and Hadoop as the communication layer, Hive as the translation layer. HadoopDB is a hybrid system of parallel database systems and MapReduce-based systems. It performance and efficient is approximate parallel database, and it also has the characteristic of scalable, fault tolerant and flexible. The inquiry of HadoopDB is expressed by SQL, and the procedure is translated into MapReduce by tools.

Sawzall [7] divides the computing processes into two stages: filtering phase and aggregation phase. The filtering phase is expressed using a procedural programming language. In this phase, data is analyzed and emitted to the aggregation phase. In the aggregation phase, aggregators process intermediate results and generate output files. Finally, these files are collected, sorting and formatting, and are merged into a single file.

DryadlLINQ [8] exploits LINQ (Language Integrated Query) to provide a powerful hybrid of declarative and imperative programming. The LINQ programming model is retained and is extend to a data parallel programming model through defining some new operations and data types. A DryadLINQ program is a sequential program com-posed of LINQ expressions. The

DryadLINQ system automatically and transparently translates the data-parallel portions of the program into a distributed execution plan which is passed to the Dryad execution platform.

DryadlLINQ has good flexibility. It can use traditional functions, modules and libraries, and it can also use standard loops to express iterative. More important, it is supported by strong tools, and it can be written and debugged using standard .NET development tools.

SCOPE [9] (Structured Computations Optimized for Parallel Execution) is a declarative and extensible scripting language proposed by Microsoft, targeted for massive data analysis. SCOPE borrows some characteristics of SQL. The data is modeled as sets of rows composed of typed columns. Some statements of SQL are retained. Users can easily define their own functions and implement their own versions of operators, such as extractors, processors, reduces and combiners. Developers can easily express problems that traditional SQL can be hardly expressed by SCOPE. The compiler and optimizer of SCOPE generate parallel execution plan. The plan is a job which is expressed by a directed acyclic graph and executed by the Dryad execution engine.

SCOPE has some advantages. 1) SCOPE is easy master, and developers who are familiar with SQL statements can use it without being trained. 2) SCOPE provides the function which is similar the SQL's view. It improves the modularity and codes reuse. It is also used to access to sensitive data and improve the security. 3) SCOPE is highly scalable. 4) SCOPE supports nested express and data conversion can be expressed as a series of simple steps.

3.4. Workflow-Oriented Parallel Programming Model

A workflow-oriented cloud computing framework (WfOC) is proposed in the paper [10]. WfOC provides a workflow-oriented cloud computing programming language. The language is composed of two parts. One part of the language is the workflow logic definition and the other part is the workflow tasks functions implementation. WfOC first analyzes workflow. It then extracts tasks from workflow and tasks are combined into an executable workflow instance based on the XML descriptor. The workflow instance is submitted to the Tasks Functions for scheduling. The Execution mode is based on MapReduce architecture. Application developers are only need to concentrate on workflow definition and function implementation of workflow tasks. They need not known complexity of distributed data and complexity of workflow tasks scheduling.

CloudWF [11] is a scalable and lightweight computational workflow system for clouds on top of Hadoop. It adopts a simple prototype workflow description language (as is shown in Figure 1) that encodes workflow blocks and block-to-block dependencies separately as standalone executable components. The workflow executing in parallel in space and time, and the management is rely on tasks scheduling and fault tolerant of MapReduce framework.

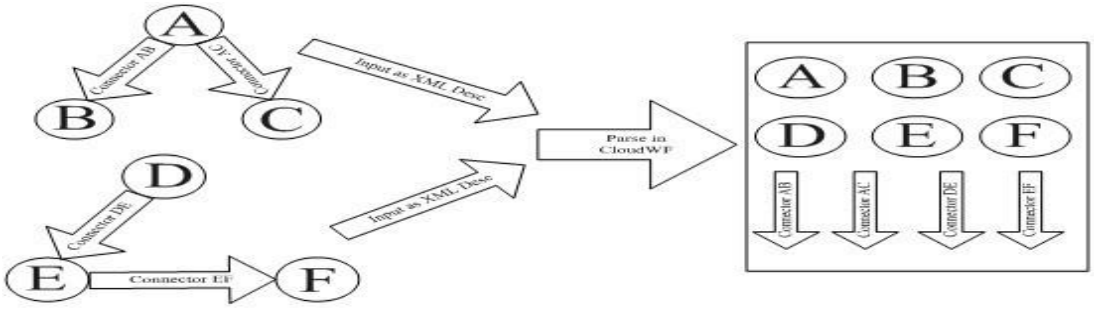

Figure 1. Breaking Up the Components of Two Workflows into Independent Blocks and Connectors

3.5. Orleans

Orleans [12] consists of three interdependent components: programming model, programming language and tools and runtime. The Orleans programming model introduces the concept of "grains". A grain consists of a single-threaded computation with its local state. All communications between grains occurs across channels. An important property of a grain is that it can migrate between computers. Migration allows Orleans to adaptively execute a system: to reduce communication latency by moving a computation closer to a client or data resource, to increase fault tolerance by moving a computation to a less tightly coupled system, and to balance the load among servers.

3.6. All-pairs

All-pairs [13] is problem is that all elements of set A are compared to all elements of set B via function F, yielding mat ix M, such that $M[i, j] = F(A[i], B[j])$. All-pairs abstraction is implemented and is used to solve the problems with all-pairs, such as biometrics, data mining and *etc.*

3.7. BSP

The Bulk Synchronous Parallel (BSP) [14] model is first proposed by Harvard's valiant which used to bridge parallel architecture and programming language. The BSP model consists of a series of super-steps. Each super-step is composed of computing phase, global communication and barrier. With the advantages in predictable performance, easily programming and deadlock avoidance. Because of these advantages, BSP model has been introduced to the cloud computing programming model.

Pregel [15] is mainly used for efficient processing large graphs. The computing processes are expressed as a sequence of iterations. In each iterative, each vertex call user-defined function, and it receives messages sent in the previous iteration, send messages to other vertices, and modify its own state and that of its outgoing edges or mutate graph topology. Pregel is similar in concept to MapReduce but much more efficient support for iterative computations over the graph. Hama [16] provides BSP library based on the Hadoop framework. It is used to processes graph by BSP model. BSPCloud [17] is a hybrid of distributed-memory and shared-memory bulk synchronous parallel (BSP) programming model. Computing tasks are first divided into a set of coarse granularity bulks which are computed by the distributed-memory BSP model, and each coarse granularity bulk is further divided into a set of bulk threads which are computed by the shared-memory BSP model.

4. Conclusions

At present, research on cloud computing programming model mainly focuses on massive data analysis. The blocks of data are independent, and the programming model is inefficiency when the data need to communicate with each other during execution. Existing programming models are hardly quantitative estimate of algorithms. This limits the development of the programming model. In addition, existing programming models cannot efficient run on the heterogeneous environment, and with little consideration of security issues. These are largely limit the development of the cloud computing programming model. This paper considers the following directions will become hotspot of the future cloud computing programming model. 1) Computing nodes massive communication is supported during the running of the program. 2) The programming model provides quantitative estimate of algorithms. 3) Load balance is supported. 4) Secure mechanism is provided. 5) Formal semantics.

Acknowledgements

This work is supported by Innovation Action Plan supported by Science and Technology Commission of Shanghai Municipality (No.11511500200).

References

[1] I. Foster, Z. Yong, I. Raicu and S. Lu, "Cloud computing and grid computing 360-degree compared", 2008 Grid Computing Environments Workshop, **(2008)**.
[2] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters", Communications of the ACM, vol. 51, no. 1, **(2008),** pp. 107-113.
[3] M. Isard, M. Budiu, Y. Yuan, A. Birrell and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks", Oper. Syst. Rev., vol. 41, no. 3, **(2007),** pp. 59-72.
[4] "Hadoop", http://hadoop.apache.org/, City.
[5] C. Olston, B. Reed, U. Srivastava, R. Kumar and A. Tomkins, "Pig latin: a not-so-foreign language for data processing", ACM, City, **(2008)**.
[6] A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz and A. Rasin, "HadoopDB: An architectural hybrid of MapReduce and DBMS technologies for analytical workloads", Proceedings of the VLDB Endowment, **(2009),** pp. 922-933.
[7] R. Pike, S. Dorward, R. Griesemer and S. Quinlan, "Interpreting the data: Parallel analysis with Sawzall", Scientific Programming, no. 13, **(2005),** pp. 277-298.
[8] Y. Yu, M. Isard, D. Fetterly, M. Budiu, G. P. K. Erlingsson and J. Currey, "DryadLINQ: A system for general-purpose distributed data-parallel computing using a high-level language", USENIX Association, City, **(2008)**.
[9] 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", Proceedings of the VLDB Endowment, **(2008),** pp. 1265-1276.
[10] P. Jinshan, C. Lizhen, Z. Yongqing and W. Haiyang, "A workflow-oriented cloud computing framework and programming model for data intensive application", Proceedings of the 2011 15th International Conference on Computer Supported Cooperative Work in Design (CSCWD), **(2011),** pp. 356-361.
[11] Z. Chen and H. De Sterck, "CloudWF: A Computational Workflow System for Clouds Based on Hadoop", Cloud Computing. Proceedings First International Conference, CloudCom 2009, **(2009),** pp. 393-404.
[12] J. Larus, "Programming Clouds, Compiler Construction", Proceedings 19th International Conference, CC, **(2010)**, pp. 1-9.
[13] C. Moretti, J. Bulosan, D. Thain and P. J. Flynn, "All-pairs: an abstraction for data-intensive cloud computing", Proceedings of the 2008 IEEE International Parallel & Distributed Processing Symposium, **(2008)**, pp. 1-11.
[14] L. G. Valiant, "A bridging model for parallel computation", Communications of the ACM, vol. 33, **(1990),** pp. 103-111.
[15] G. A. Malewicz, H. Matthew, J. C. Bik Aart, H. llan, L. Naty and C. Grzegorz, "Pregel: A system for large-scale graph processing", Proceedings of the 2010 International Conference on Management of Data, **(2010)**, pp. 135-145.
[16] S. Sangwon, Y. J. Edward, K. Jaehong, J. Seongwook, K. Jin-Soo and M. Seungryoul, "HAMA: An efficient matrix computation with the MapReduce framework", The 2nd IEEE International Conference on Cloud Computing Technology and Science, **(2010)**, pp. 721-726.
[17] X. Liu, W. Tong, F. ZhiRen and L. WenZhao, "BSPCloud: A Hybrid Distributed-memory and Shared-memory Programming Model", International Journal of Grid and Distributed Computing, vol. 6, no. 1, **(2003),** pp. 87-97.

