An Investigation on Scheduling Policies for Cloud-based Software Systems

Jia Ru
Department of Computing
The Hong Kong Polytechnic University
Hong Kong SAR
Email:csrjia@comp.polyu.edu.hk

***Abstract*—**
***Background*: The rapid diffusion of cloud computing technology has been a focus of interest for enterprises due to its higher scalability and availability and greater elasticity. Nevertheless the limited scheduling mechanisms for running applications in the cloud have been a major challenge.**
***Aim*: This project introduces an effective scheduling algorithm, which attempts to maximize cloud resources utilization, improve the computation ratio, and reduce makespan, overhead and delay in a cloud-based software system.**
***Method*: (1) Analyze different scheduling algorithms which can be adopted in cloud-based systems and simulate these algorithms using CloudSim. (2) Evaluate these algorithms performance and determine both of their advantage and disadvantage. (3) Propose an improved scheduling algorithm or policy and verify the proposed algorithm in CloudSim as well as extend CloudSim. (4) Study MapReduce framework and grasp its operating principles. (5) Analyze MapReduce scheduling algorithms. (6) Propose and optimize an efficient scheduling algorithm on MapReduce.**
***Conclusion*: Proposed scheduling policies should effectively in improving the number of completed tasks, reduce costs, and promote the development of scheduling environment. In the premise of processing accuracy, improved MapReduce scheduling policy can improve the utilization rate of resources, reduce workloads of nodes and optimize management of resources.**

***Index Terms*—Cloud Computing, CloudSim, MapReduce, Scheduling, Software Metrics**

I. Introduction

With further development and research on cloud computing, there are some interesting research challenges and problems to be overcome, such as data centre network structure expansibility, energy conservation, replica policies and scheduling mechanism [1] [2] [3] [4]. The primary objective of this project is optimizing scheduling polices. The aim of cloud computing is to realize cooperation work and resources sharing. However, the resources in cloud-based systems are heterogeneous, dynamic and diversified due to different client requirements. This makes management of resources very complex. Adopting an appropriate scheduling policy and optimizing scheduling mechanisms to improve the performance and utilization of resources are a significant research area in this regard. In traditional distributed environment, the aim of optimizing scheduling is mainly focusing on system performance, such as system throughput, CPU utilization rate and almost never considering QoS (Quality of Service). In this project, I am not only emphasizing resource utilization rate and system performance, but also requiring a guaranteed QoS of users based on different demands. In the premise of guaranteed resource utilization rate, scheduling policies mainly focus on allocation management of resources and satisfy all the resource demands of users. This project will propose scheduling policies that can effectively improve performance guarantees such as task completion rate, as well as reduce overall costs. Consequently, considering some specific criteria (cost, completion time, etc.) or priorities of tasks and resources, I will propose an effective scheduling algorithm, which will maximize the cloud resource utilization, improve the computation ratio, and reduce the makespan and delay in the cloud-based software system.

Nowadays, MapReduce as a distributed processing model and a high-efficiency task scheduling model is designed and developed by Google [5]. It is a useful tool to process data-intensive jobs. However, the default MapReduce scheduling algorithms have some drawbacks. For example, in a MapReduce cluster, there only exists one JobTracker used to allocate jobs to TaskTrackers. JobTracer takes charge of the entire system scheduling. Once JobTracker fails or crashes, the compute nodes will not be able to complete. On the other hand, if a greater number of jobs are submitted by hundreds of clients and a large amount of TaskTrackers are distributed in clusters, then JobTacker will face heavy working pressure and workload. Hence the efficiency of the entire system will be reduced due to the execution capability of the JobTracker. Moreover, if the network bandwidth of JobTracker is lower than normal, then the system performance will also be decreased. To solve the problem of single JobTracker node failure, this project will propose a new scheduling policy, which can be considered as a pluggable component for MapReduce. The proposed scheduling algorithm involves more than one JobTrackers distributed in the cluster to manage resources and schedule jobs to TaskTrackers.

II. Literature review

A. Concepts of Cloud Computing

Cloud computing is based on the concept of infrastructure convergence [6] and sharing services [7], in an attempt to provide unique types of services through provisioning of dynamically scalable and virtualized resources. It is a com-

bination of techniques from parallel computing, distributed computing, as well as platform virtualization technologies.

B. Scheduling model in Cloud Computing

Scheduling is a decision process, and its content is deploying resources to applications of different clients at a suitable time, or during a specific period of time.

1) ***Cloud computing scheduling model****:* Cloud computing scheduling model is mainly constructed by Client, Broker, Resources, Resources supporter and Information Service. Fig.1 shows the scheduling model structure [8]. The tasks implemented can be divided into serial application, parallel application, parameter scan application, cooperation application and so on. Different clients use resources at different prices. Broker is a middle interface between clients and resources. It is used to find resources, choose resources, accept tasks, return scheduling results, and exchange information. Broker supports different scheduling policies, which can allocate resources and schedule tasks in accordance to the demands of clients. Broker is constituted by Job Control Agent, Schedule Advisor, Explorer, Trade Manager and Deployment Agent.

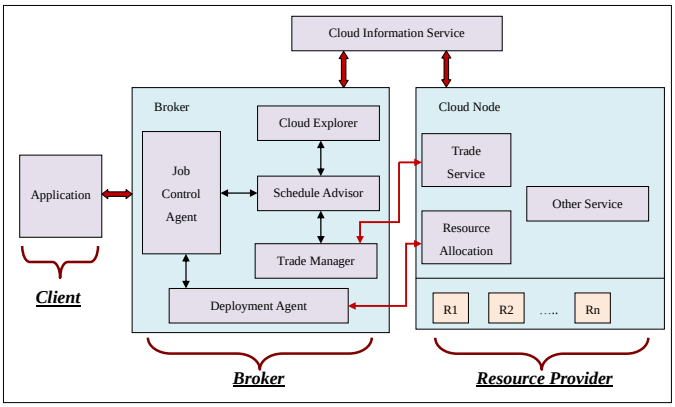

Fig. 1. Chart of scheduling model structure

- Job Control Agent: It is responsible for monitoring jobs in the software system, such as schedule generation, jobs creation, status of jobs and communicating with clients and schedule advisor.
- Schedule Advisor: It is used to determine resources, allocate available resources which satisfy the demands of clients such as deadline and cost, and to assign jobs.
- Cloud Explorer: It communicates with cloud information service to find resources, identifies the list of authorized machines and records resources status information.
- Trade Manager: It determines resources access cost and tries to communicate with resources at a low cost under the guidance of schedule advisor.
- Deployment Agent: It uses scheduler instruction to activate the execution of tasks and update the status of tasks execution to Job Control Agent in regular intervals.

2) ***Basic scheduling methods****:* Scheduling methods always consider two aspects: one is characteristics of tasks, and the other is characteristics of datacenter resources [9]. There are two possibilities of submission tasks on resources. One is tasks submission on the resources where the input data is available. The other is submission on some specific resources based on some criteria [10].

3) ***Resource allocation****:* The resources in the cloud computing can be allocated in different ways. Traditional method of task scheduling in cloud environment uses the direct tasks of clients as the overhead application base [11]. Considering maximum utilization rate of resources, the algorithms for resources allocation can be FCFS (First Come First Service), SJF (Shortest Job First), priority scheduling, RR (Round Robin) scheduling, random, greedy, Genetic Algorithm [12], Ant Algorithm and other heuristic scheduling methods [13].

C. Scheduling mechanism

1) ***Scheduling policy based on replica****:* Replica is a hot topic in cloud computing, which makes up single-point failure of storage object, bad fault tolerance, bad access performance and so on. K-means algorithm [14] is a typical dynamic clustering algorithm which modifies iteration point-to-point. Its principle is base on quadratic sum function.

2) ***Online scheduling problem****:* According to different scheduling time, task scheduling can be divided into two models: batch scheduling and online scheduling. In online scheduling, as long as one task arrives, it is scheduled to a free time resource to perform immediately. If all the resources are busy, the task needs to wait. In existing cloud computing environment, online scheduling policy mainly focuses on distribution management of resources, and satisfies various demands of clients. However, it does not pay enough attention to supporting service. According to different part given online, online scheduling can be classified into 4 models: scheduling jobs one by one, unknown running time, jobs arrive over time and interval scheduling [15].

3) ***Batch scheduling problem****:* In batch scheduling, when a task arrives, the task does not gain service immediately and is collected in a task set. During some special time or after an event, all the tasks can be scheduled to resources. Min-min is a typical batch scheduling algorithm. The concept of Min-min is allocating the best resource to the task which computing capabilities are minimum. It can improve system throughout and enable task sets to gain minimum completion time.

4) ***MapReduce****:* MapReduce system partitions input data and schedules the execution of programs in clusters of commodity machines. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key. The functional model with user-specified map and reduce operations allows us to parallelize large computations easily and to use re-execution as the primary mechanism for fault tolerance [2] [5].

- FIFO (First In First Out) Scheduling algorithm:

JobTracker uses FIFO scheduling algorithm to schedule MapReduce tasks of clients, and then puts all the MapReduce tasks into a queue. Finally, according to tasks submission time order, tasks can be selected to execute. Though, FIFO

scheduling algorithm is simple but it ignores the different demands of different tasks. Hence, this makes the execution performance and utilization of resources lower.

- Fair Scheduling algorithm:

It is firstly proposed by Facebook. It supports classification of tasks and allocates different types of resources to the various types of tasks in order to improve performance quality. Moreover, it can dynamically appraise the number of parallel tasks to make full use of resources. The fair scheduler organizes jobs into pools, divides resources fairly between these pools and also allows assigning guaranteed minimum shares to pools. By default, there is a separate pool for each user, so that each user gets an equal share of the cluster. It is also possible to set a job's pool based on a configurable attribute, such as user name and unix group. Within each pool, jobs can be scheduled using either fair sharing or FIFO scheduling [16]. However, this scheduling policy does not consider actual workload of the task nodes and it results in workload unbalance. At runtime, the actual workload of a task node is determined by resource consumption of the running tasks rather than number of those.

- Capability Scheduling algorithm:

It is proposed by Yahoo. It supports parallel operations of multi tasks and dynamically allocates resources to increase the efficiency of tasks execution [16]. The Capacity Scheduler is designed to allow sharing a large cluster while giving each queue a minimum capacity guarantee. When a task is submitted, it is put into a queue. Each queue gets some TaskTracker resources based on configuration to process Map and Reduce operations. Available resources can be dynamically allocated to the queues with heavy workloads. In a queue, tasks can be operated according to their different priorities. High-level priority task will be executed first, but Capacity Scheduler does not support preempting priority. Nevertheless, this scheduling algorithm cannot automatically set up configuration of queues and also cannot choose queues by itself.

III. Methodology

The project mainly consists of the following 2 phases:

A. Phase 1: Analyzing different scheduling policies and improving some scheduling algorithms

In the initial phase, I will study different scheduling policies such as Greedy, Round Robin, Genetic Algorithm, Ant Algorithm, priority scheduling and other heuristic algorithms. Next, I will evaluate these algorithms performance and determine their advantage, disadvantage as well as applicability.

For simulating all these different algorithms, I will adopt CloudSim framework. CloudSim is a new generation and extensible simulation platform which enables seamless modeling, simulation, and experimentation of emerging Cloud computing infrastructures and management services to be accomplished [7] [17]. CloudSim is used to verify the correctness of the proposed algorithm. CloudSim toolkit is used to simulate heterogeneous resource environment and the communication environment. The layered CloudSim architecture mainly contains 4 layers: SimJava, GridSim, CloudSim and User code [17]. Through conducting an empirical experiment to examine various different scheduling algorithms that are important to the development of software for the cloud platforms, I will propose an improved scheduling algorithm and test as well as verify the performance of proposed algorithm in CloudSim. I also need to extend CloudSim simulator for our experimentation which enables some parameters and data to be varied easily.

Research Progress and revenant work:

- Evaluate and analyze existing algorithms

I have evaluated some existing scheduling algorithms, such as job grouping-based scheduling, bandwidth-aware job grouping-based scheduling, greedy scheduling algorithm, and sequential algorithm (FIFO). Fig.2 shows the total processing time of different algorithms. We can obviously see that sequential scheduling algorithm processing time is largest and processing time of greedy algorithm is smallest when the number of cloudlets is changed.

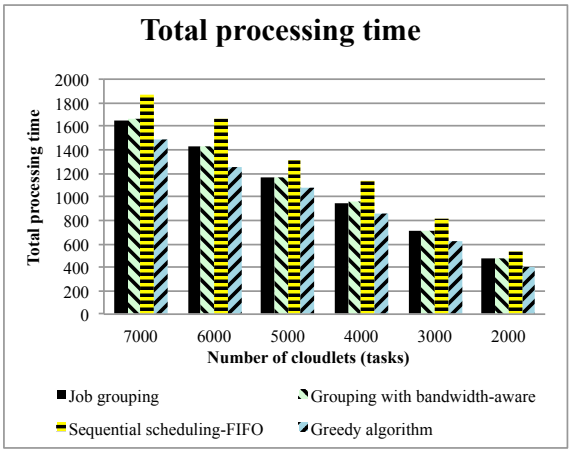

Fig. 2. Total processing time of different scheduling algorithms

- The proposed algorithm

I have proposed an algorithm which integrates with the three algorithms: task grouping, prioritization (bandwidth-aware) and SJF (Shortest-Job-First). Compared with traditional grouping-based scheduling algorithm, the proposed algorithm is suitable for both very lightweight jobs and the tasks with random and unpredictable processing requirements. It also considers the communication and transmission rate of resources to reduce the transmission latency. Integrated SJF can reduce tasks waiting time. Fig.3 shows the total processing time of different number of tasks. Fig.4 presents average waiting time of different number of tasks. In comparison with existing task grouping algorithms, results show that the proposed algorithm waiting time significantly decreased over 30% and processing time decreased by 7.25%. The proposed method effectively minimizes waiting time and processing time and reduces processing cost to achieve optimum resources utilization and minimum overhead, as well as to reduce influence of bandwidth bottleneck in communication.

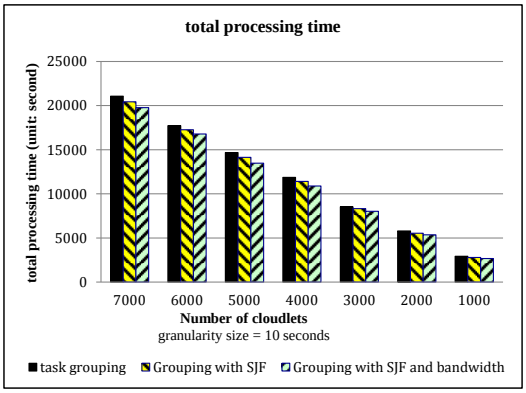

Fig. 3. Total processing time with different number of tasks

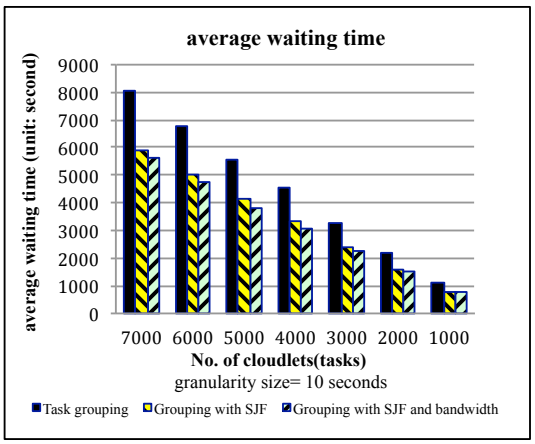

Fig. 4. Average waiting time with different number of tasks

B. Phase 2: Analyzing MapReduce scheduling algorithms and improving or optimizing MapReduce scheduling policies

In this phase, for investigating MapReduce framework and grasping its operating principles, I will conduct some empirical experiments on Hadoop platform. Hadoop is open source of Google MapReduce [16]. A Hadoop cluster consists of normal PCs as computing nodes and owns great horizontal scalability and high expandability. Hadoop architecture mainly contains 4 components: Client, JobTracker, TaskTracker and Hadoop Distributed File System (HDFS). I will use 6 PCs to set up a cluster, where one PC is as a master and the rest are as slaves and clients. Afterwards, I will run scheduling algorithms in Hadoop and evaluate different scheduling deployment performance as well as analyze these scheduling policies' advantage and drawbacks. Finally, I will propose or optimize an efficient MapReduce scheduling algorithm.

IV. Discussion and Expected Results

The proposed algorithm is an attempt to reduce network delay, maximize the utilization of cloud resources and simultaneously, achieve a minimum waiting time. In addition, it is able to determine deadline constraints in providing Quality of Service (QoS). The improved MapReduce scheduling policy should increase the availability and reduce nodes load as well as optimize resource management.

In phase 1, I focus on evaluating the performance and feasibility of different existing scheduling algorithms. Due to the limitations of time and cost efficiency, adopting a simulated cloud environment to carry out some empirical experiments is needed. It is reliable and efficient to appraise the performance and applicability of my proposed algorithm on CloudSim under simulation of different cloud deployments. Although all the hypotheses are close to real, accurate capability and applicability of scheduling policies in real cloud-based software systems still cannot be certain. It is more meaningful that the outcomes of this project can be applied in practice. MapReduce is a widely applied cloud program framework in IT industries. Therefore, in Phase 2, I emphasize scheduling policies on MapReduce. The proposed scheduling policy should be evaluated on CloudSim first, and then be embedded on MapReduce. The expected results can be considered as pluggable component for MapReduce to be applied in practice.

References

[1] A. Fox, R. Griffith *et al.*, "Above the clouds: A berkeley view of cloud computing," *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Tech. Rep. UCB/EECS*, vol. 28, 2009.

[2] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Grid Computing Environments Workshop, 2008. GCE'08*. IEEE, 2008, pp. 1–10.

[3] C. Germain-Renaud and O. F. Rana, "The convergence of clouds, grids, and autonomics," *Internet Computing, IEEE*, vol. 13, no. 6, pp. 9–9, 2009.

[4] B. Leiba, "Having one's head in the cloud," *Internet Computing, IEEE*, vol. 13, no. 5, pp. 4–6, 2009.

[5] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[6] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 50–55, 2008.

[7] R. Buyya, C. 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.

[8] R. Buyya, D. Abramson, J. Giddy *et al.*, "An economy driven resource management architecture for global computational power grids," in *Proceedings of the 2000 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2000)*, 2000, pp. 26–29.

[9] V. Korkhov, J. Moscicki, and V. Krzhizhanovskaya, "Dynamic workload balancing of parallel applications with user-level scheduling on the grid," *Future Generation Computer Systems*, vol. 25, no. 1, pp. 28–34, 2009.

[10] S. Singh and K. Kant, "Greedy grid scheduling algorithm in dynamic job submission environment," in *Emerging Trends in Electrical and Computer Technology (ICETECT), 2011 International Conference on*. IEEE, 2011, pp. 933–936.

[11] S. Selvarani and G. Sadhasivam, "Improved cost-based algorithm for task scheduling in cloud computing," in *Computational Intelligence and Computing Research (ICCIC), IEEE International Conference on*, 2010.

[12] S. Sivanandam and S. Deepa, *Introduction to genetic algorithms*. Springer Publishing Company, Incorporated, 2007.

[13] M. Choudhary and S. Peddoju, "A dynamic optimization algorithm for task scheduling in cloud environment," *International Journal of Engineering Research and Applications*, vol. 2, pp. 2564–2568, 2012.

[14] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, *The R*-tree: an efficient and robust access method for points and rectangles*. ACM, 1990, vol. 19.

[15] J. Sgall, "On-line scheduling," *Online algorithms*, pp. 196–231, 1998.

[16] T. White, *Hadoop: The definitive guide*. O'Reilly Media, 2012.

[17] R. Calheiros, R. Ranjan, C. De Rose, and R. Buyya, "Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services," *arXiv preprint arXiv:0903.2525*, 2009.