Volume 3, Issue 12, December 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Dynamic Load Balancing Algorithm for Cloud Computing using Mobile Agents

Nandita Goyal[*]
Information technology Department,
ABES Engineering College, Ghaziabad, India

Abstract—The Cloud Computing concept offers to increase the capacity or add capabilities dynamically without investing in new infrastructure, training new personnel, or licensing new software. Gigantic-scale heterogeneous distributed computing environments (like Clouds and Computational Grids) can easily access the vast amount of computing resources at very economical cost. The primary advantages of cloud computing systems are high performance, availability, and extensibility at low cost. To realize these benefits, however, system designers must overcome the problem of allocating the considerable processing capacity available, so that it is used to its fullest advantage. In this paper, we propose an approach for implementing dynamic load balancing mechanism for cloud computing environment using mobile agents.

Keywords—Cloud computing, Mobile agents, Dynamic Load Balancing, Parallel Processing, Distributed Computing

I. Introduction

Cloud computing relies on sharing of resources to achieve coherence and economies of scale, similar to a utility over a network. At the foundation of c loud computing is the broader concept of converged infrastructure and shared resources. The cloud also focuses on maximizing the effectiveness of the shared resources. However, there are a number of technical challenges that need to be tackled before these benefits can be fully realized, which include system reliability, resource provisioning, and efficient resources consuming etc[2]. Among them, load-balancing is the most important issue that needs to be addressed, in order to increase the service level agreement (SLA) and improve utilization of the resources. Cloud resources are usually not only shared by multiple users but are also dynamically reallocated per demand. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any one of the resources. Using multiple components with load balancing instead of a single component may increase reliability through redundancy. The concept of load balancing is not new and has been around for a long time in the field of distributed systems. In its most abstract form, the problem of load balancing is defined by considering a number of parallel machines and a number of independent tasks, each having its own load and duration [1].

We are interested in dynamic load balancing schemes which seek to minimize total execution time of a single application running in parallel on a multicomputer system. To do so, an optimal trade off between the processing and communication overhead and degree of knowledge used in the balancing process must be sought. Dynamic Load-Sharing Mechanism [2] can provide several advantages like reducing the time required to complete a task, reducing the network traffic, and increasing the throughput for the system. In this paper, we propose an approach for implementing dynamic load-sharing mechanism using mobile agents in cloud computing environment which will improve the overall efficiency of system by trying to minimize the service response times and hence data transfer cost by redirecting requests to an optimal remote server that is chosen in terms of communication latency and workload. This paper includes 5 sections to describe the approach. Section II describes different load balancing mechanisms. In section III, the approach for implementation of dynamic load balancing mechanism in cloud computing environment using mobile agents is described. Section IV describes the steps in execution for the approach and algorithms for agents. Section V concludes the paper with a discussion of our results and future work.

II. EXISTING LOAD BALANCING ALGORITHMS

A. Static Load Balancing Algorithm

In this method, the performance of the nodes is determined at the beginning of execution. Then depending upon their performance the workload is distributed in the start by the master node. The slave processors calculate their allocated work and submit their result to the master. A task is always executed on the node to which it is assigned that is static load balancing methods are non-pre-emptive. A general disadvantage of all static schemes is that the final selection of a host for process allocation is made when the process is created and cannot be changed during process execution to make changes in the system load[4]. Major load balancing algorithms are Round Robin[5] and Randomized Algorithms[6], Central Manager [7]Algorithm and Threshold[4,8] Algorithm.

B. Dynamic Load Balancing Algorithm

Dynamic algorithms have the potential to outperform static algorithms by using system-state information to improve the quality of their decisions. Unlike static algorithms, dynamic algorithms allocate processes dynamically when one of the processors becomes under loaded. Instead, they are buffered in the queue on the main host and allocated dynamically upon requests from remote hosts[4]. This method is consisted of Central Queue Algorithm and Local Queue Algorithm[9].

III. IMPLEMENTATION APPROACH

Motivated by the promising technology of mobile agent framework [10], we develop an approach for the implementation of Dynamic load-sharing mechanism [11] for cloud computing environment using mobile agents.

A. Mobile Agent Framework

To solve a large problem, multiple Java mobile agents [12] are launched to designated hosts on the cloud computing environment, where they perform their tasks. Each agent supports the basic tasks of communication and synchronization by assuming the role of a process in a parallel processing application. Threads are used inside each agent that strengthens concurrent execution of an application. First, a leader (main) agent is created which dynamically creates and dispatches a variable number of mobile agents to several workstations-hosts.

B. Dynamic Load Balancing Mechanism

In dynamic load sharing, the system attempts to find the best host for an incoming task at the run time [11]. Decisions are made using the information on the current system states. However, this approach requires gathering and maintaining of the system state information, because to achieve load sharing, a snapshot of the global loading information in a specific time frame must be acquired. The load sharing mechanism is applied at the server group. We require that each member server is capable of handling individual request and they have partial overlap in their functionality and information. Three major components are required to design dynamic load balancing mechanism namely, the server module agent (SMA), the load information agent (LIA), and the job dispatching agent (JDA).

Server Module Agent: Server Agent Module is a stationary agent that is incorporated in server part of the system. It waits for the client requests and serves them. It uses the concept of threshold policy to identify overload condition.

Load Information Agent: This is a mobile agent that carries the global load information from one server to another. Each server stores its own copy of load information with LIA. Timestamps are used to synchronize the information collected from different servers.

Job Dispatching Agent: This is a mobile agent that is created by SMA to trigger the job selection and migration process when a server becomes overloaded. On activation JDA carries jobs from reallocation list to the appropriate server based on global load information.

C. Steps in Execution

The entire framework can be divided into two parts- static part which is responsible for general coordination, creation and manipulation of agents and dynamic part which is responsible for load balancing. The steps required for the execution of this approach are as follows-

- The Leader agent is created.
- It creates and dispatches Load Information agent and other parallel agents.
- The Leader agent dispatches itself to the appropriate host (server).
- The SMA in server part accepts the incoming requests.
- Leader agent creates a Task_Handler object and initiates it in a new separate thread.
- The server site stores its own copy of load information with the Load Information agent.
- SMA keeps a check on the global load information with LIA. If the load value increases the predefined threshold value T, SMA creates JDA to trigger job selection/migration process. JDA carries the jobs in the job reallocation list to the appropriate remote servers for execution.
- The Leader agent requests Task_Handler to perform a specific task. While waiting for the first Task_Handler to complete its tasks, additional Task_Handler objects may be launched to accomplish other tasks.
- Finally, the leader agent collects the results from other parallel agents and returns them to user.

D. Algorithms

{Nomenclature: L= Local load with each SMA, GL= Global Load Information with LIA, T= selected Threshold}

Algorithm: Server Module Agent

```
Begin:
    1.  SMA triggers the LIA and each SMA stores its L with LIA.
    2.  If (L>T)
        Create a reallocation list and trigger JDA.
        Else goto step 3.
    3.  Execute the request in job queue.
    4.  Stop
End.
```

Algorithm: Load Information Agent

```
Begin:
    1. LIA travels to each SMA and updates the values of L and GL.
    2. If (JDA is triggered by SMA)
       LIA updates the L information of that SMA and provides GL value to JDA to select a server for reallocation.
    3. Stop
End.
```

{Nomenclature: Ss= Selected Server, Ls=Load at selected server}

Algorithm: Job Dispatching Agent

```
Begin:
    1. JDA checks the value of GL from LIA to select a server for reallocation.
    2. JDA carries the last job from reallocation list to the Ss.
    3. If (Ls=T)
       Goto step 1.
       Else reallocate the job.
    4. Stop.
End.
```

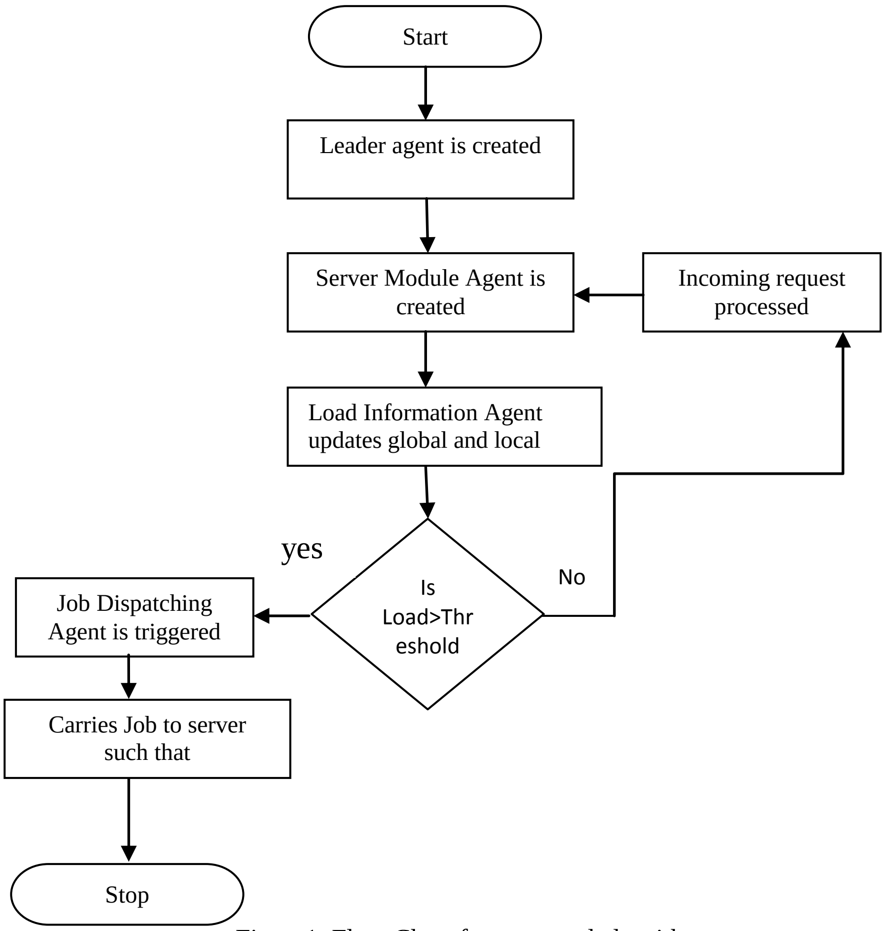
Start
Leader agent is created
Server Module Agent is created
Incoming request processed
Load Information Agent updates global and local
Is Load>Threshold
yes
No
Job Dispatching Agent is triggered
Carries Job to server such that
Stop

Figure1. Flow Chart for proposed algorithm

IV. CONCLUSION

In this paper we have proposed an approach for implementation of Dynamic Load Balancing mechanism [11] with Cloud computing environment using mobile agent framework [10] as discussed earlier. This approach will improve the overall performance and increase the throughput of the system. The time required to complete a task by using mobile agents for Dynamic Load Sharing mechanism is significantly less. This improvement comes in Dynamic Load Sharing mechanism, as it tries to divide the task of a node according to load of other nodes in the network. This way almost all segments of task finish their execution in same time

REFERENCES

[1]. Eddy Caron , Luis Rodero-Merino "Auto-Scaling , Load Balancing and Monitoring in Commercial and Open-Source Clouds " Research Report ,January2012

[2] Rich Lee,Bingchiang Jeng"Load balancing Tactics in Cloud"2011 International Conference on Cyber Enabled Distributed Computing and Knowledge Discovery

[3]. Karimi Abbas, Zarafshan Faraneh “A New Fuzzy Approach for Dynamic Load Balancing Algorithm” International Journal of Computer Science and Information Security, Vol. 6, No. 1, 2009

[4]. S. Sharma, S. Singh, and M. Sharma, "Performance Analysis of Load Balancing Algorithms," World Academy of Science, Engineering and Technology, Vol. 38, 2008.

[5]. Z. Xu and R. Huang, "Performance Study of Load Balancing Algorithms in Distributed Web Server Systems” CS213 Parallel and Distributed Processing Project Report.

[6]. R. Motwani and P. Raghavan, "Randomized algorithms," ACM Comput. Survey, Vol. 28, pp. 33-37, 1996.

[7]. P. L. McEntire, J. G. O'Reilly, and R. E. Larson, Distributed Computing: Concepts and Implementations. New York: IEEE Press, 1984.

[8]. W. I. Kim and C. S. Kang, "An adaptive soft handover algorithm for traffic-load shedding in the WCDMA mobile communication system," presented at WCNC'2003, 2003.

[9]. W. Leinberger, G. Karypis, and V. Kumar, "Load Balancing Across Near-Homogeneous Multi-Resource Servers," presented at s, Cancun, Mexico, 2000.

[10]. C. Panayiotou, G. Samaras, E. Pitoura, and P. Enripidou,1999, Parallel Computing Using Java Mobile Agents, IEEE Transactions on Network, pp. 430-436.

[11]. J. Cao, X. Wang, S.K. Das, 2001, A Framework of using Cooperating Mobile Agents to achieve Load Sharing in Distributed web server groups, ScienceDirect Vol 20, issue 4, PP 591-603.

[12]. D. Staneva, P. Gacheva, Building Distributed Applications with Java mobile agents, International Workshop NGNT.