Mini Project Report

Agent Based Dynamic Resource Allocation on Federated Clouds

Submitted in partial fulfillment of
the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

by

Roll No	Name
B090924CS	Alok Saw
B090904CS	Shubhangam Agrawal
B090006CS	Stein Astor Fernandez
B090930CS	Sunil Kumar Suthar

Under the guidance of
Mr. Saidalavy Kaladi

Department of Computer Science & Engineering
National Institute of Technology, Calicut
Kerala - 673601

Department of Computer Science & Engineering

National Institute of Technology Calicut

Certificate

This is to certify that the project work entitled "**Agent Based Dynamic Resource Allocation on Federated Clouds**", submitted by Alok Saw (B090924CS), Shubhangam Agrawal (B090904CS) and Stein Astor Fernandez (B090006CS) and Sunil Kumar Suthar (B090930CS) to National Institute of Technology Calicut towards partial fulfillment of the requirements of the award of Degree Of Bachelor of Technology in Computer Science and Engineering, is a bonafide record of the work carried out by them under my supervision and guidance.

Project Guide
Mr. Saidalavy Kaladi

Place : Calicut
Date :

Abstract

Current large distributed systems allow users to share and trade resources. In cloud computing, users purchase different resources like network bandwidth, computing power and storage system from one or more cloud providers for a limited period of time with a variable or fixed price. Federated cloud is a mechanism for sharing resources thereby increasing scalability. An improved algorithm has been devised in which a multi-agent system consisting of Consumer Agents, Broker Agents and Resource Provider Agents work in tandem to provide the consumer with the desired resources with maximum efficiency. In this system, the user need not know where the resources reside or who is providing them.

Agent Based Dynamic Resource Allocation on Federated Clouds

Shubhangam Agrawal
Department of Computer Science & Engineering
National Institute of Technology, Calicut
shubhangam@gmail.com

Alok Saw
Department of Computer Science & Engineering
National Institute of Technology, Calicut
alok.saw123@gmail.com

Stein Astor Fernandez
Department of Computer Science & Engineering
National Institute of Technology, Calicut
steinfernandez@gmail.com

Sunil Kumar Suthar
Department of Computer Science & Engineering
National Institute of Technology, Calicut
faultysunil@gmail.com

Abstract—**Current large distributed systems allow users to share and trade resources. In cloud computing, users purchase different resources like network bandwidth, computing power and storage system from one or more cloud providers for a limited period of time with a variable or fixed price. Federated cloud is a mechanism for sharing resources thereby increasing scalability. An improved algorithm has been devised in which a multi-agent system consisting of Consumer Agents, Broker Agents and Resource Provider Agents work in tandem to provide the consumer with the desired resources with maximum efficiency. In this system, the user need not know where the resources reside or who is providing them.**

Keywords-MARA, JADE, Cloud Computing

I. Introduction

A number of computer researchers and practitioners have attempted to define a Cloud in various ways. A proposed definition is as follows: "A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service level agreements established through negotiation between the service provider and consumer [1]".

In cloud computing, hardware and software are services. So, providers need QoS in allocation of resources to Cloud consumers. Cloud providers and consumers are self interested parties who will try to maximize their benefits. Allocation of resources to Cloud consumers is a complex procedure due to the complexity of optimal allocation of resources (efficient allocation with limited resources and maximum benefit). The price of the resources in a cloud is determined dynamically based on a demand-supply model or statically based on a fixed price model [2], [3].

We can solve the resource allocation anomaly by using a multi-agent system [4]. In multi-agent system, providers and consumers are agents. The resources are distributed amongst several agents and these agents will perform the allocation process. This method is generally known as Multi-agent Resource Allocation (MARA) [5]. A wide range of issues like network routing, Grid computing [6] can be solved using MARA. The multi-agents are sometimes called "social welfare agents" [5], that is, they will give maximum welfare (benefits) to providers and consumers.

In federated clouds [2], [3], users request more than one type of resources from different providers. So they need the information about all service providers and the status of each provider. Choosing the best provider is very difficult because they don't know the dynamic price of each resource in different clouds.

We have implemented the multi-agent system described in the IEEE paper "Agent Based Dynamic Resource Allocation on Federated Clouds" [7], and in the course of implementation, made several optimizations and enhancements to the working algorithm. The platform used to implement the system was JADE[8]. JADE stands for Java Agent DEvelopment Framework. It is a software framework fully implemented in Java language. It simplifies the implementation of multi-agent systems through a middle-ware that complies with the FIPA[9] specifications and through a set of graphical tools that supports the debugging and deployment phases.

II. Agent Based Dynamic Resource Allocation

In negotiation based resource allocation, the consumer communicates with the resource provider directly. If the requested resources in their entirety are available with one provider, this works well. However, in a federated cloud, by definition, the resources are spread across multiple providers. In this case, aggregating resources from across multiple providers will prove to be a daunting task. Moreover, based on supply, demand and consumer feedback, resource prices are constantly fluctuating. This also needs to be accounted for, making resource allocation a complex and laborious undertaking.

A. *Working of the Multi-Agent System*

Fig.1 shows the proposed system having three agent types: Resource Broker Agents, Consumer Agents and Resource Provider Agents.

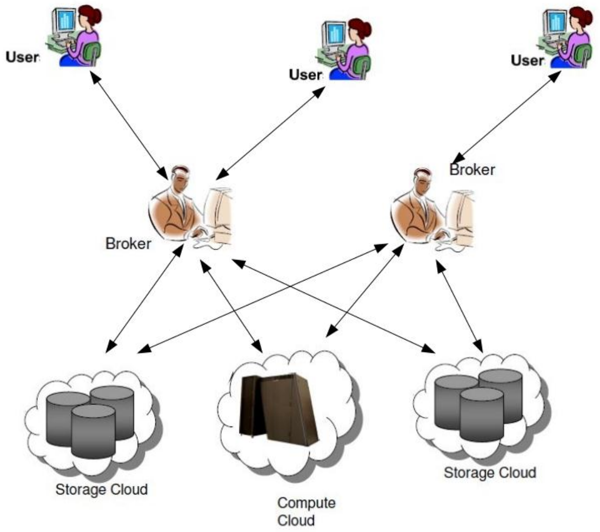

Fig.1. System Model

The Consumer Agent sends its request for resources to a Resource Broker Agent. The Resource Broker Agent collects and stores information about the resources provided by the various Resource Provider Agents. Whenever a Resource Provider Agent updates its available resources, the Resource Broker Agent is notified so it can maintain up-to-date information in its database. The Resource Broker Agent negotiates the requests of the Consumer Agent with the Resource Provider Agent. If the request is not fulfilled, it repeats the process with another Resource Provider Agent. The Resource Broker Agent maintains a quality rating of the resources provided by each Resource Provider Agent based on the feedback received from the Consumer Agents who have purchased those resources.

B. *Problem Modelling*

The attributes are as follows:

rc Resource requested
pl Duration for which the resource is requested
qty Quantity of the resource requested
pr(O) Proposed Price
dr Price-demand ratio
inU The amount of that resource currently in use
QoS Quality of Service

C. *Formulae for calculating Feedback and QoS*

After a Consumer Agent completes its task successfully using the resources allocated to it by a Resource Provider Agent, it provides feedback to the Resource Broker Agent based on the quality of the service provided. The Resource Broker Agent maintains the quality rating of each resource provider as an Exponential Weighted Moving Average.

Utility = pl / Time Taken for Task (1)

QoS = α x Feedback + (1-α) x Old QoS (2)

Cost = (1+Demand Ratio) x Price (3)

III. NEGOTIATION PROTOCOL

In this system, we use three protocols for negotiation between the Consumer Agent, Resource Broker Agent and Resource Provider Agent, as well as a task processing algorithm. The following algorithms along with the accompanying illustration will detail the intricacies of the protocols used for communication between the agents.

A. *Consumer Agent Algorithm*

Begin
1. Initialize
2. Send CFP <rc, pl, qty> to RBA
3. Receive PROPOSE (Cost) from RBA
4. If cost within COST_LIMIT then
    a. Send ACCEPT_PROPOSAL to RBA and go to step 5

   Else

    b. Send REJECT_PROPOSAL to RBA and go to step 3
5. If REFUSE received then go to step 2
6. Receive INFORM from RPA else go to step 3
7. Send CONFIRM to RPA to accept agreement and mark time.
8. Receive CONFIRM
9. Calculate feedback by equation (1)
10. Send INFORM <feedback> to RBA

End

B. *Resource Broker Agent Algorithm*

Begin
1. Initialize
2. Receive INFORM <update> from RPA
3. If entry is present:
    a. Update the entry

   Else

    b. Add the entry
4. Receive CFP from CA
5. Select the best provider from provider list on the basis of quality
6. Calculate Proposed cost = quantity * cost * pl
7. Send PROPOSE <Proposed cost> to CA

8. If ACCEPT_PROPOSAL from CA then
    a. Forward CFP to RPA <CA, Proposed cost>

   Else

    b. Receive REJECT_PROPOSAL from CA and go to step 5
9. Receive response from RPA
10. If PROPOSE then
    a. Negotiation completed; go to step 13

    Else

    b. Update cost of resources in provider list by getting INFORM from RPA
    c. receive REFUSE from RPA
    d. Go to step 6
11. Receive feedback from CA
12. Update provider list with feedback

End

C. Resource Provider Agent Algorithm

Begin

1. Initialize
2. Send INFORM <rc, qty, cost> message to RBA with resource info
3. Receive CFP from RBA
4. If price not acceptable then
    a. Send current costs by INFORM
    b. Send REFUSE to RBA and go to step 3

    Else

    c. Send PROPOSE to RBA
    d. Send agreement INFORM to CA
    e. Receive CONFIRM from CA
    f. Increase demand-price ratio
    g. Add task to task queue

End

D. Task Processing Algorithm

1. Maintain FIFO queue of tasks
2. Check if enough resources are allocatable; if so run task.
3. On task completion, (after pl) send CONFIRM to CA.
4. Remove task from Task queue.
5. Reduce demand-price ratio.

On initialization, the Resource Provider Agent sends an INFORM message to the Resource Broker Agent containing details regarding the resources it provides i.e. quantity, price, price-demand ratio, and amount currently in use. The Resource Broker Agent receives this message and checks if the entry is already present. If so, it updates it; else it adds the entry. Once the Consumer Agent is initialized with the resources that it is intended to purchase, it sends a CFP to the Resource Broker Agent with the name of the resource, length of time for which it is needed and the quantity required. On receiving the CFP, the Resource Broker Agent finds the appropriate resource in its database and selects the best provider from the list on the basis of Quality of Service of that provider. It calculates the cost of purchasing the required quantity of that resource for the required time, and sends a PROPOSE message to the Consumer Agent. If the Consumer Agent accepts, the Resource Broker Agent forwards the CFP to the pertinent Resource Provider Agent. If the Consumer Agent does not accept, then the Resource Broker Agent selects the next most suitable Provider Agent from the list, and so on.

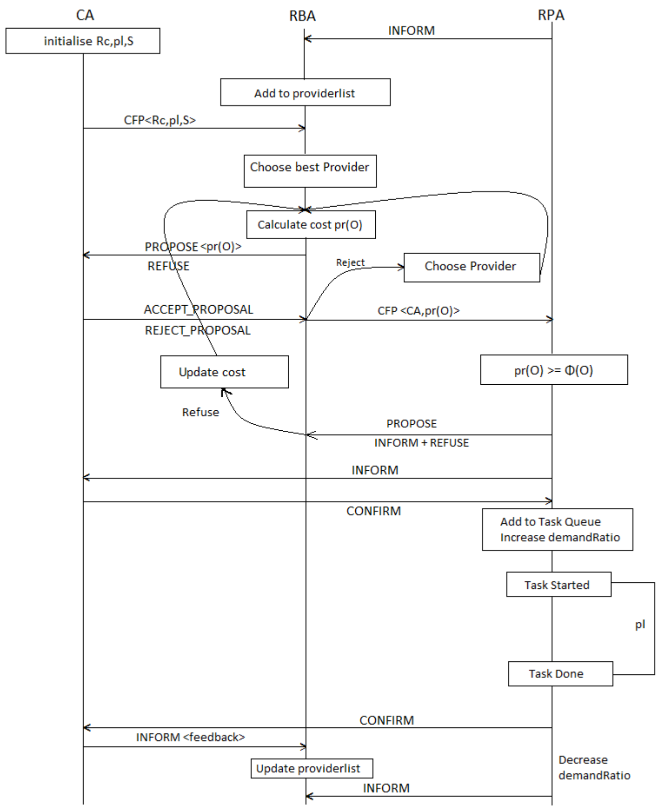

Fig.2. Data Flow Diagram

If the Resource Provider Agent in question finds the cost acceptable, then it will send a PROPOSE message to the Resource Broker Agent, which ends the role of the Broker in this negotiation. If it does not find the cost acceptable, it sends an INFORM message with the updated cost of the resource to the Resource Broker Agent, which recalculates the resource cost and repeats the corresponding steps.

Having accepted the CFP, the Resource Provider Agent sends an INFORM message to the Consumer Agent and receives a CONFIRM message in response. The Provider Agent then increases its demand-price ratio accordingly. The Consumer Agent marks the start time of the task as well. The task is added to the task queue, which is run as a FIFO queue based on availability of resources. Once the task is completed, a CONFIRM message is sent to the Consumer Agent indicating completion. The Consumer Agent then calculates Utility using the equation (1) and sends it to the Broker Agent via an INFORM message. The Broker Agent then recalculates the Quality of Service of the Provider Agent in question and updates the corresponding value in its provider list.

IV. IMPLEMENTATION

The project has been implemented entirely in JAVA, using the JADE framework [2]. The Resource Broker Agent has been coded in the file BrokerAgent.java, the Consumer Agent in ConsumerAgent.java and the Resource Provider Agent in ProviderAgent.java. The GUI is coded in the file ProviderGUI.java.

A snapshot of the inter-agent communication is presented in the file Sniffer.png using the Sniffer agent present in JADE.

Compilation can be achieved by running ./make.sh.

To run, type ./run.sh. This will start a GUI with one Broker Agent initialized. To add agents, select the Create New Agent option.

- **Resource Broker Agent**

  AgentName: [as desired]

  ClassName: BrokerAgent

- **Resource Provider Agent**

  AgentName: [as desired]

  ClassName: ProviderAgent

  This will start a GUI to add available resources.

- **Consumer Agent**

  AgentName: [as desired]

  ClassName: ConsumerAgent

  Arguments: cost_limit, rc, pl, qty

V. CONCLUSION AND FUTURE WORKS

This paper proposes a model for dynamic resource allocation in federated clouds. The model acts as an improvement over existing methods already developed. In this model three types of agents are used, namely, Consumer Agent, Resource Brokering Agent and Resource Providing Agent. The Resource Brokering agent contains all information about resources and it allocates resources to Consumer Agent from Resource Providing Agent.

The main advantage of this model is that consumer need not bother about where the resources are placed and what is its cost. Consumer can get resources with minimum cost. We have implemented the protocol model and tested the system using JADE.

Future improvements include enabling the Consumer Agent to requisition more than one resource which will be allocated over multiple Resource Provider Agents. The system can also be implemented on a cloud simulator in order to test real-world performance.

ACKNOWLEDGMENT

We would like to thank Mr. Saidalavy Kaladi for consenting to be our project guide and Mr. Sreenu Naik B for being the project co-ordinator and their continued support and encouragement throughout the duration of the mini project.

REFERENCES

[1] R. Buyya, C.S. Yeo, and S. Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on, pages 5–13. IEEE, 2008.

[2] M. Mihailescu and Y. Teo. Strategy-Proof Dynamic Resource Pricing of Multiple Resource Types on Federated Clouds. Algorithms and Architectures for Parallel Processing, pages 337–350, 2010.

[3] M. Mihailescu and Y.M. Teo. Dynamic Resource Pricing on Federated Clouds. In 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, pages 513–517. IEEE, 2010.

[4] Y. Shoham and K. Leyton-Brown. Multiagent systems: Algorithmic, game-theoretic, and logical foundations. Cambridge Univ Pr, 2008.

[5] Y. Chevaleyre, P.E. Dunne, U. Endriss, J. Lang, M. Lemaıtre, N. Maudet, J. Padget, S. Phelps, J.A. Rodrıguez-Aguilar, and P. Sousa. Issues in multiagent resource allocation. Special Issue: Hot Topics in European Agent Research II Guest Editors: Andrea Omicini, 30:3–31, 2006.

[6] SS Manvi and MN Birje. An agent-based resource allocation model for grid computing. In Services Computing, 2005 IEEE International Conference on, volume 1, pages 311–314. IEEE, 2005.

[7] Haresh M V, Saidalavi Kalady and Govindan V K, Agent Based Dynamic Resource Allocation on Federated Clouds

[8] Java Agent DEvelopment Framework, http://jade.tilab.com

[9] The Foundation for Intelligent Physical Agents (FIPA) is a body for developing and setting computer software standards for heterogeneous and interacting agents and agent-based systems.