Agent Based Dynamic Resource Allocation on Federated Clouds

Haresh M V*, Saidalavi Kalady† and Govindan V K‡
*Department of Computer Science and Engineering
National Institute of Technology Calicut
Email: hareshmv@gmail.com
†Department of Computer Science and Engineering
National Institute of Technology Calicut
Email: said@nitc.ac.in
‡Department of Computer Science and Engineering
National Institute of Technology Calicut
Email: vkg@nitc.ac.in

***Abstract*—Current large distributed system allows 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. Allocating resources in cloud is a complex procedure. In order to improve resource allocation agent based resource allocation method is used. In this method, the user needs not know who is the cloud service provider and where the resources reside. The consumer gets the resources with the minimum price. The proposed system has three types of agents namely Consumer agent, Resource Brokering agent and Resource Provider agent.**

Keywords: MAS, MARA, Auction, Negotiation

I. Introduction

A number of computer researchers and practitioners have attempted to define 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 inter connected 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 maximizing 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 being distributed amongst several agents and these agents will do 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 multiagents are some times called "social welfare agents"[5], that is it 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 resources in different clouds. In this project we suggest a broker based multi-agent system.

II. Literature Survey

Yann Chevaleyre et. al.[5] has defined the issues of multiagent resource allocation.The main issues are protocol used for communication, strategies of producer consumer agents and the algorithms which are used for optimal resource allocation.The protocol may be a centralized mechanism like auctions or distributed mechanism like negotiation. Marian Mihailescu et. al.[2] has come up with a dynamic pricing strategy. In dynamic pricing, price of the resources are set according to the forces of demand and supply. They suggest an auction with a third party called market makers, who collect the bids and select the winner and compute the payment.

In multi agent system several market based methods are used for resource allocation. Shneidman et. al.[7] specifies how markets could solve resource allocation problem and the challenges on existing market based allocation. Rich Wolski et. al.[8] specifies the strategies for computational grid. They describe the efficiency of resource allocation under two different market conditions like commodity markets and auctions. They compared both market strategies in terms of price stability, market equilibrium, consumer efficiency and producer efficiency. Xindong e.t al.[9] proposed a cloud resource allocation based on market mechanism. In this method, if the allocation is in equilibrium state under the price of each resources then it is optimal. It will take the benefit of consumer and provider to the peak. They use a genetic algorithm[9] for price adjusting.

Wei-Yu Lin et. al.[10] proposed a dynamic auction mechanism for resource allocation on cloud systems. They used the second priced auction mechanism[10] with dynamic price. It assures that the provider can get reasonable profit and efficient allocation of its computational resources. Another type of auction is the continuous double auction[11]. In this method, provider determines the value of request by its workload and consumer determines the bid value based on the remaining time for bidding and the remaining resources for bidding.

Bo An et. al.[12] proposed resource allocation by negotiation with decommitment penalty. Consumer negotiates with two providers and on agreement with one, the other one will have to be given a de-commitment penalty. This method uses strategies like: *Deadline* - consumer must satisfy its resource requirements by the deadline, *Seller's cost* - provider/seller will not accept a price lower than its cost. In this mechanism consumers directly communicate with the providers.

III. Agent Based Dynamic Resource Allocation

In Negotiation based resource allocation method, consumer and provider directly communicate with each other. If the requested resources are available in a single cloud provider it is very simple. If it is in a federated cloud, resources will have to be collected from different providers, which is a complex procedure. Also price of each resource dynamically varies based on demand and supply. So getting current price of resources from different providers is very difficult.

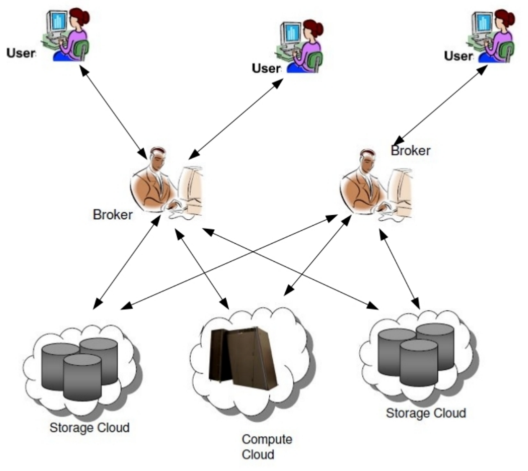

Fig. 1. System Model

Figure 1 shows the proposed system, it consist of some brokers as mediators. The proposed system has three types of agents namely Consumer Agent, Resource Brokering Agent[13], and Resource Provider Agent. The Consumer Agent sends its request to Broker Agent. The Broker Agent contains all information about cloud providers like where is the location, what is the current price of each resource, lowest cost of the resources and the provider which provides high quality of service. The Broker Agent assigns a grade to the providers based on the feedback from the consumers to which it has done the allocations. All the time the Broker Agent will be checking the status of each of the cloud providers. The Broker Agent negotiates with Resource Provider Agents and if the requirement of a Consumer Agent is not fulfilled by a single provider, it initiates a negotiation with another Resource Provider Agent.

A. Problem Modelling

Consumer Agent has the following attributes:

- ***Rc*** - set of resources requested by the consumer. For each $r \in Rc$, *q(Rc,r)* is the quantity of resource *r*[12].
- ***est*** - earliest start time of the task to be done. The task should not start before *est*[12].
- ***pl*** - resource usage period length, the amount of time for which the resource is requested for[12].
- ***dlt*** - deadline time of the task. The task must be started before *dlt*, otherwise it will be considered as failed[12].
- ***s*** - optional attribute for the consumer to specify a specific service provider.
- ***st*** - task start time.

Resource Brokering Agent has a *providerlist* which contains the information about each resources, its cost and its quality of service. The resource brokering agent periodically updates the *providerlist*.

Cost of a resource r is *c(r)*. The total cost of *Rc* is $\sum_{r \in Rc}$ *q(Rc,r)c(r)pl(c)*[12]. The Demand price ratio of resource r is $\psi(r)$. So the expected price of resource set *Rc* at RPA is:

$$\phi(O) = \sum_{r \in Rc} q(Rc,r)c(r)(1+\psi(r))pl(c)[12] \qquad (1)$$

Utility of a Consumer Agent depends on the task completion time and its payment. The task completion time is higher if the task start time is closer to the earliest start time (*est*) and lower when closer to deadline time (*dlt*). The payment is calculated by dividing the quantity of resources by the cost of resources. Let opl represent the actual amount of time taken by the task. CA's utility at time t is:

$$u_c(t) = \frac{dlt - st}{dlt - est} + \frac{pl}{opl} + \frac{q(Rc)}{c(Rc)} \qquad (2)$$

B. Negotiation Protocol

In this model three protocols are used for negotiation, between CA, RBA and RPA. Algorithm 1 is the protocol used for negotiation of CA with other two, Algorithm 2 for negotiation between RBA and the other two and Algorithm 3 for RPA. Figure 2 shows the data flow digram of protocol model.

Algorithm 1 is the protocol used for negotiation of CA with other two, Algorithm 2 for negotiation between RBA and the other two and Algorithm 3 for RPA.

Figure 2 shows the data flow digram of protocol model. Consumer agent initialize the CFP message. CA send its CFP message to RBA. RBA extract the resources from the request and it searches each resource in provider list based

Algorithm 1: Consumer Agent Communication

```
begin
    1. initialize Rc,est,dlt,pl,S
    2. send CFP $\langle Rc, est, dlt, pl, S\rangle$ to RBA
    3. receive PROPOSE(Cost of Rc c(Rc)) from
        RBA // Cost of Rc
    4. if cost acceptable then
        4.1. send "ACCEPT_PROPOSAL" to RBA
    else
        4.2. send "REJECT_PROPOSAL( cost-limit)"
        to RBA
        4.3. goto step 3
    end
    5. receive PROPOSE from RBA
    6. if confirmed then
        6.1. send "AGREE" to RBA
    else
        6.2. send "REFUSE" to RBA
        6.3 goto step 5
    end
    7. receive "CONFIRM" from RPA
    8. accept agreement and send "INFORM" to RPA
    9. running task ...
    10. calculate utility by eq(1)
    11. send feedback "INFORM" to RBA
end
```

Algorithm 2: RBA Communication

```
begin
    1. receive CFP $\langle Rc, est, dlt, pl, S\rangle$ from CA
    2. Extract resource from Rc
    3. Select the best provider from providerlist
    4. Calculate cost of each resources in Rc
    5. send "PROPOSE(c(Rc))" to CA
    6. receive ack from CA
    7. if "ACCEPT_PROPOSAL" then
        7.1. send CFP $\langle Rc, est, dlt, pl\rangle$ to RPA
    else
        7.2. receive
        "REJECT_PROPOSAL(cost-limit)" from CA
        7.3. goto step 3
    end
    8. receive response from RPA
    9. if PROPOSE then
        9.1. send "PROPOSE" request to CA
    else
        9.2. update cost of resources in providerlist
        9.3. goto step 4
    end
    10. receive ack from CA
    11. if "AGREE" then
        11.1. send "CONFIRM" to RPA
    else
        11.2. send "REFUSE" to RPA
        11.3. goto step 3
    end
    12. receive feedback "INFORM" from CA
    13. update providerlist with feedback
end
```

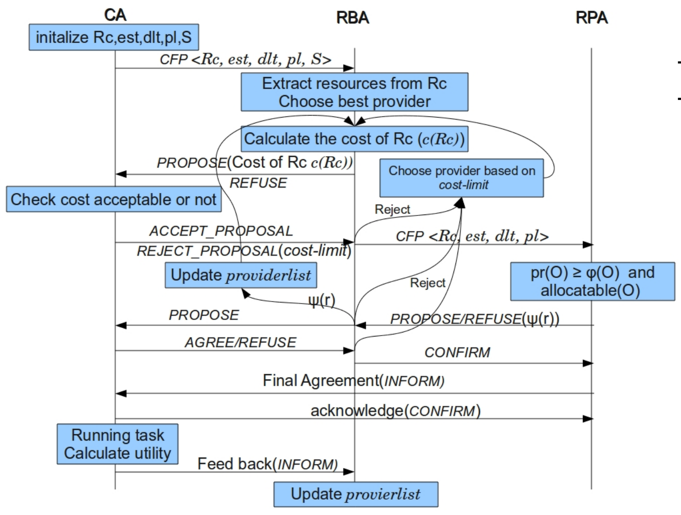

Fig. 2. Data flow diagram

Algorithm 3: RPA Communication

```
begin
    1. receive CFP O from RBA
    2. if pr(O) $\geq \phi(O)$ then
        2.1. if allocatable(O) then
            2.1.1. send "PROPSE" to RBA
        else
            2.1.2. send "REFUSE" to RBA
        end
    else
        2.2. send current cost of Rc to RBA
    end
    3. receive confirm from RBA
    4. if "CONFIRM" then
        4.1. send agreement "INFORM" to CA
        4.2. receive confirm "CONFIRM" from CA
    end
end
```

Algorithm 4: Resource Allocation[12]

```
begin
    1. calculate revenue of each requests by $pr(O)$ - $cost(O)$
    2. sort requests in descending order based on revenue
    3. pick one by one until no requests
    4. if request O fulfils then
        4.1. return true
    else
        4.2. return false
    end
end
```

on, maximum QoS and minimum unit cost. After RBA gets the resource list, it calculates the total cost of resource set Rc by $\sum_{r \in Rc} q(Rc,r)c(r)pl(c)$ and send to Consumer Agent by PROPOSE message. If the cost is acceptable CA sends the ACCEPT_PROPOSAL to RBA. If the cost is higher than expected cost CA sends the expected cost of resources by REJECT_PROPOSAL.

RBA receive the ACCEPT_PROPOSAL message from CA, then initializes a CFP message and sends to Resource Provider Agent. RPA receives the CFP message and extracts the resources and its total cost. Then calculate the expected cost of requested resources by equation(1). If the expected cost is less than or equal to CFP cost then accept the request and run the resource allocating algorithm for requested resource that is available at that time. If it is available for assignment, send PROPOSE message to RBA. If the expected cost is higher than requested cost then send the current demand/price ratio to RBA by REFUSE message. RBA update the provider list with this demand price ratio and repeat the same procedure.

If RBA receives PROPOSE message from RPA then send the partial agreement to CA by PROPOSE message. If the agreement acceptable, CA send AGREE message to RBA and RBA send the CONFIRM message to RPA for agreement confirmation. If the agreement is not acceptable, CA send a refuse message to RBA and repeat the protocol from the start. When RPA get the confirmation, it sends the final agreement to CA by CONFIRM message. And CA sends the acknowledge by INFORM message. After reaching the final agreement, the consumer agent begins the execution. After completion of the task, CA calculates the utility of the resources and sends it as feedback to RBA. RBA updates the provider-list based on the received value.

IV. 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 being employed.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 aim at implementing the protocol model and testing the system using JADE[14]. Then only we can compare with other related approaches for performance evaluation.

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] J. Shneidman, C. Ng, D.C. Parkes, A. AuYoung, A.C. Snoeren, A. Vahdat, and B. Chun. Why markets could (but don't currently) solve resource allocation problems in systems. In *Proceedings of the 10th conference on Hot Topics in Operating Systems-Volume 10*, page 7. USENIX Association, 2005.

[8] R. Wolski, J.S. Plank, J. Brevik, and T. Bryan. Analyzing market-based resource allocation strategies for the computational grid. *International Journal of High Performance Computing Applications*, 15(3):258–281, 2001.

[9] X. You, X. Xu, J. Wan, and D. Yu. RAS-M: Resource Allocation Strategy Based on Market Mechanism in Cloud Computing. In *ChinaGrid Annual Conference, 2009. ChinaGrid'09. Fourth*, pages 256–263. IEEE, 2009.

[10] W.Y. Lin, G.Y. Lin, and H.Y. Wei. Dynamic Auction Mechanism for Cloud Resource Allocation. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 591–592. IEEE, 2010.

[11] H. Izakian, B.T. Ladani, K. Zamanifar, A. Abraham, and V. Snasel. A continuous double auction method for resource allocation in computational grids. In *Computational Intelligence in Scheduling, 2009. CI-Sched'09. IEEE Symposium on*, pages 29–35. IEEE, 2009.

[12] B. An, V. Lesser, D. Irwin, and M. Zink. Automated negotiation with decommitment for dynamic resource allocation in cloud computing. In *Proceedings of the nineth international joint conference on autonomous agents and multi-agent systems*, pages 981–988. Citeseer, 2010.

[13] G. Haring, G. Kotsis, A. Puliafito, and O. Tomarchio. A transparent architecture for agent based resource management. In *Proceedings of the International Conference on Telecommunications (ICT'98)*, pages 338–342. Citeseer.

[14] Java Agent DEvelopment Framework . http://jade.tilab.com/. [Online; accessed 20-Nov-2010].