Middle-East Journal of Scientific Research 24 (6): 1949-1955, 2016
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2016.24.06.23576

Profit Maximization Using Knapsack Algorithm for Resource Allocation in Cloud

[1]Anitha Jaganathan, [2]E. Iniya Nehru and [3]N. Ananthi

[1]M.E Software Engineering, Easwari Engineering College, Chennai, India
[2]Senior Technical Director National Informatics Centre, Chennai, India
[3]Information Technology, Easwari Engineering College, Chennai, India

Abstract: Cloud computing allows the customers to dynamically change their requirements on the pay-as-you-use manner. The SLA is an agreement normally negotiated and accepted between the service provider and user. The provisioning of resources can be either lower bound or upper bound, which results in the violation of SLA or wastage of resources and money respectively. In order to overcome such defects an algorithm is used to calculate the number of virtual mahines to satisfy all tenants service level agreement based on which resources can be provisioned dynamically. This paper proposes an algorithm that gives results in the maximization of profit to service provider. The integer linear programming and the greedy-heuristic based approaches are used for the provisioning of resources to the clients. The algorithm is generated on the basis of software as a service. In saas perspective, virtual machines are considered as resources. Therefore, the virtual machines are provisioned by the provider to satisfy the dynamic needs of the clients. In some cases, the clients are served earlier for many reasons like they contain short job and minimum deadline. In these cases, some of the virtual machines can be unused, for utilizing the unused virtual machines they are involved in Auction.

Key words: Auction and Resource Allocation

INTRODUCTION

Cloud computing helps sharing of resources based on user demand. Cloud resources are shared by multiple users and can be reallocated to users. The availability of high-capacity networks, low-cost computers and storage devices as well as the distributed adoption of hardware virtualization, service-oriented architecture and autonomic and utility computing have led to a growth in cloud computing.

In Cloud Computing the elasticity in the resource provisioning is one of the major challenges for the providers. Scheduling and resource are important in cloud computing because service provider should allocate the needed resources to the needed tenants, so that all the tenants are served. The SLA is an agreement normally negotiated and accepted between the service provider and the user. The static SLA based resource provisioning is not efficient for the real life situation, in which the user tends to change their needs dynamically, which makes the introduction of dynamic SLA.

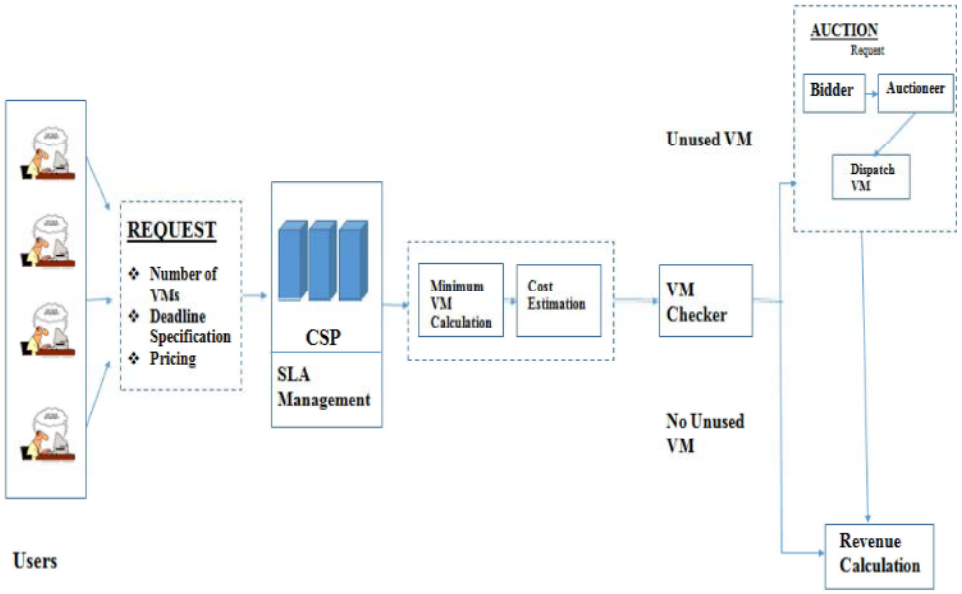
The main intention of a Service provider in cloud is to completely avoid the penalty and to maximize their profit. The penalty has to be paid when Service Provider violates the SLA. The Client specifies their constraints in SLA. After negotiating SLA, the Service Provider must satisfy the Client's requirements as possible it can, otherwise, they have to pay the penalty to their Clients. In this project, SaaS service model is implemented where virtual machine are considered as resources which are allocated on the basis of slot. Therefore, before negotiating SLA, certain algorithms have been applied to take informed decision to avoid the penalty. With the minimum virtual machine calculation algorithm, Service Provider can determine total number of virtual machine required for over all slots and with this Provider can analyze whether they could satisfy their Client's or not. If not, it is better to avoid that particular Client and serve up other Client's. In some slots, the Client's work will be finished earlier, so some of the virtual machine can go unused. These virtual machine can be given in to auction in order to maximize the revenue of the Provider's and also for efficient resource management.

Corresponding Author: Anitha Jaganathan, M.E Software Engineering, Easwari Engineering College, Chennai, India.

Related Works: Bipin B.Nandi [1] proposed the heuristic based resource management technique in which heuristic for maximum tenant on-boarding problem results in unutilized resources in some slots. The Integer Linear Programming (ILP) and the Greedy-heuristic Based approaches are used for the provisioning of resources to the clients. Massimiliano Rak [2] proposed a simulation based approach by which fast prediction can be done. Here the SLA life cycle can be categorised as Negotiation phase, which describes about what the SLA should provide. Monitoring phase, which checks whether the SLA is respected or not. Enforcement phase, in which the implementation and the provisioning of resources are done. Zuling Kang [3], proposed a new resource allocating algorithm namely, CRAA/FA (Cloud Resource

Allocating Algorithm via Fitness-enabled Auction), deals with these new challenges in cloud resource allocation. CRAA/FA models the cloud platform as the resource market, while the cloud resource providers and renters as sellers and buyers, respectively. During the allocation period, buyers submit bids and sellers submit requests at any time during the trading period. At any time when there is a bid and request that matches or is compatible with a price, a trade is executed immediately. Dimitri P. Bertsekas [4], proposed auctioning algorithms for solving network flow problems. Nonlinear network flow problems are convex minimization problems that arise in many applications from transportation to finance. The network structure of the constraints and the separation of context make this class of problems suitable for special algorithms that are orders of magnitude which makes it faster than standard convex optimization methods. Dual coordinate algorithms such as -relaxation can be used as the best methods for solving such separable convex cost network flow problems. Duo Liu [5], proposed a light weight method of SLA management for Cloud Computing. Cloud computing is a promising trend for computing and data storage based on its high scalability and efficiency. Cloud providers should provide resources with reliability and stability. Establishing a cloud platform must be easy to deploy and use for developing an enhanced monitoring tool to detect SLA violations at the application layer is not only important to customers, but also critical to cloud providers as well.

System Architecture: This architecture aims to satisfy the dynamic SLA and enhance the profit of the provider to involve the unused virtual machine in the auctioning. The Tenants specify their requirements in the form of SLA to the service provider. These SLA is forwarded by the Service Provider to the Minimum VM Calculation that gives the result in minimum number of virtual machines which able to satisfy the requirements of all the Clients. This is given to Tenant Maximization to find the number of virtual machines that are not used. These virtual machines are involved in auction to enhance the profit of the Service Provider. The revenue is calculated at last by satisfying the tenants and bidders.



Use Techniques: The measurement of upper bounded value of virtual machine must be calculated to find out the number of unused virtual machine for each slot k. This process is start from the minimum virtual machine calculation. Here the SLA is an input. The dynamic SLA of Tenant is represented as < M_{ij},D_i,Q_{ij}, F_{ij},G_{ij}, T_i > where M_{ij} is j^{th} mode of i^{th} Client. (eg) M_{12} represents mode 2 of client 1. Q_{ij} is license for j^{th} mode of i^{th} client. F_{ij} is job percentage for j^{th} mode of i^{th} client and G_{ij} is price for j^{th} mode of i^{th} client.This is represented as shown in the below table.

Table I: SLA Specification

Client No	C1		C2		C3			C4	
Deadline	2		3		4			2	
Mode	1	2	1	2	1	2	3	1	2
License	3	5	4	8	4	8	10	4	7
% Job Completed	20	50	30	55	20	30	40	25	50
Price	40	90	35	60	30	40	70	30	50

Here T_i can be considered as critically constraint.
[1] This is expressed as (d,m) where $d \leftarrow \{1, 2,.. . D_i\}$ and $m \leftarrow M_i$.

(eg) $T_i = \{(1, m_{12}), (2, m_{12})\}$.

For slot 1 Tenant 1 wants to execute the mode 2 and for slot 2 also client 1 wants to execute the mode 2.

Intuitively, this expresses the slot allocation constraints that the tenant might have, in addition to the guarantee of completion within the specified deadline. In order to satisfy the critically constraints of the tenant, the below mentioned condition should hold.

$x_{ijk} = y_i \square (k, j) \square T_i$, $1 \leq i \leq P$ (P is number of Tenants) where X_{ijk} and Y_i are binary variables for

$1 \leq i \leq P$, $1 \leq j \leq |M_i|$, $1 \leq k \leq D_i$.

Minimum VM Calculation: The Measurement of Upper bounded value of the virtual machines is essential [1], because that results lead to calculate the wastage of resources and money. This algorithm namely Minimum VM Calculation will calculate the required number of that will barely meet the SLA with the critical constraints of the tenants. If the critical constraint of the tenant is not satisfied then that particular tenant is not allocated at all. Here the importance is given to the critical constraints.

Algorithm 1: Minimum VM Calculation (MVC)

1. **Procedure Min_VM** (SLA)
2. **for** i=1 to P **do**
3. Set k=D_i and job=0
4. **While** k>0 **do**
5. **If** (k,j) ◎ T_i **then** //critical slot
6. Allocate mode j at slot k
   Update L_k+=Q_i(j), job+=F_i(j) and $R^{rem}_{i,Di-k}$
   K=k-1
7. **Else**
8. Pick M_i for which $F_i(j) \geq R^{rem}_{i,Di-k}$
   Update L_k+=Q_i(j), job+=F_i(j) and $R^{rem}_{i,Di-k}$
9. **if** job ≥ 100 **then**
10. Update y_i =1
11. **else**
12. k=k-1
13. **End if**
14. **End if**
15. **End while**
16. **End for**
17. **return** $\max_k L_k$
18. **End procedure**

Here [1] L_k be the total virtual machines required for slot *k*. Initially L_k = 0 for all *k* and $R^{rem}_{i,k}$ be the average percentage of completion required for client C*i* after executing its job in previous *k* slots.

Initially $R^{rem}_{i,0} = \frac{100}{Di}$ for all *i*.

The clients are allocated for the slots with the consideration of the critical constraint. The slots allotted as per the algorithm is tabulated in Table II.

Table II: Minimum VM Calculation By the Algorithm

SLOTS	SLOT 1	SLOT 2	SLOT 3	SLOT 4
Selected Modes	m12 (5)	m12 (5)	--	--
	m21 (4)	m21 (4)	--	--
	m31 (4)	m31 (4)	m22 (8)	--
	m42 (7)	m42 (7)	m32 (8)	m32 (8)
VM Needed	20	20	16	8

The total number of VM needed for slot 1,2,3,4 are 20,20,16,8 respectively. The Maximum of virtual machines among the slots is 20.

Maximum Tenants Calculation: These calculated virtual machines will be given as input for the second algorithm namely, Tenant Maximization, in which the tenants are allocated for the needed resources. After the allocation of resources, some virtual machines are left unutilized in some slots. This algorithm will return those numbers of virtual machines present in the particular slots and the cost acquired by allocation of resources to the tenants till now. This revenue is added with the generated revenue by auction.

Algorithm 2: Tenant Maximization

1. Procedure Max_Tenant (SLA, V)
2. Sort the Tenants according to the deadline as P*
3. **foreach** i as P*
4. **if** (k,j) ◎ T_i **then** //critical slot
5. Allocate mode j at k
6. Update $F_{i,k}^{rem}$ += $F_{i,k}^{rem}$, $cost_i$+=$G_i(k)$ and $L_{i,k}^{rem}$ += $L_{i,k}^{rem}$.
7. **else**
8. Set k=1
9. **while** k ≤ D_i **do**
10. Sort $F_i(j)$ in descending order, in which M_i* is the sorted mode
11. Choose first mode M_i* for which $F_i(j) \leq F_{i,k-1}^{rem}$ and $Q_i(j) \leq L_{k,i-1}^{rem}$
12. **if** no such mode exist or k=D_i **then**
13. Pick M_i* for which $Q_i(j) \leq L_{k,i-1}^{rem}$ and $F_i(j) \leq F_{i,k-1}^{rem}$
14. Update Fi, $cost_i$, $L_{k,i-1}^{rem}$
15. **end if**
16. **if** $F_{i,k-1}^{rem} \leq 0$ **then**
17. Update y_i=1
18. **else**
19. k=k+1
20. **End if**
21. **End while**
22. **End if**
23. **End foreach**
24. **return** $\sum_{i=1}^{n} costi$
25. End procedure

Here [1] $L_{k,i-1}^{rem}$ be the remaining virtual machines available at the k^{th} slot after considering all the tenants up to the $(i-1)^{th}$ tenant.

Intuitively, $L_{k,i-1}^{rem}$ is basically the quantity L-$\sum_{i'=1}^{i-1} Q\,i'(j)Xi'jk$. $F_{i,k-1}^{rem}$ is be the remaining percentage of completion for tenant C_i's job after the $(k-1)^{th}$ slot.

$F_{i,k-1}^{rem}$ basically has the quantity 100 – $\sum_{k'=1}^{k-1} Fi(j)Xijk'$. Initially for all *I*, $F_{i,0}^{rem}$ =*100* and for all *k*, =L.

After the successful allocation of the tenants and the balance virtual machines left unused in slot 1,2,3,4 are 0,0,6,10 respectively. These unused virtual machines are given for auction. The slots tabulated by this algorithm can be viewed below in Table III.

Table III: Slot Allocation By Algorithm

SLOTS	SLOT 1 (20)	SLOT 2 (20)	SLOT 3 (20)	SLOT 4 (20)
Selected Modes	m12 (5)	m12 (5)	--	--
	m42 (7)	m42 (7)	--	--
	m22 (8)	m21 (4)	m21 (4)	--
	--	m31 (4)	m33 (10)	m33 (10)
Balance VM	**0**	**0**	**6**	**10**

Auctioning the Virtual Machines: The balance virtual machines are handled by the third algorithm namely, Auctioning the virtual machines, which is implemented for each slots which has unused virtual machines. Initially, the provider splits the unused virtual machines in slot as, 90% of virtual machines is allotted for normal auction method and remaining 10% of virtual machines is allotted for emergency constraint. First, in normal auction method, the bidders (tenants) will specify their bid consisting number of l virtual machines required and price per vm. Before determining the winners in auction, minimum price quoted and maximum virtual machines required by the bidders are verified in bidder selection method. If there is any virtual machines remained unused after the auction, then it is given for the constraint method. Then, later emergency constraint is checked. If it is applicable, then cost is computed or otherwise those virtual machines also given for normal auction method. From the algorithm 2, slot 1 and 2 doesn't contain any unused virtual machines, but slot 3 has 6 and slot 4 has 10 unused virtual machines. Then these virtual machines are given for the auction method.

- For slot 3, it has 6 virtual machines. 5 virtual machines are given for normal auction and 1 remaining virtual machine is given to constraint method. The bidder's specification for auction is given below in Table IV.

Table IV: Auction Table With Price Per VM By Bidders For Slot3

SLOTS/BIDDERS	VM REQUIRED	PRICE PER VM
B1	7	20
B2	5	18
B3	12	21
B4	8	24

Table V: Auction Table For Selected Bidder With Price Per VM For Slot3

SLOTS/BIDDERS	VM REQUIRED	PRICE PER VM
B2	5	18

Based on the auctioning the licenses algorithm the selected bidder is bidder 2 and cost acquired for 5 virtual machines in auction is 90.

- For slot 3, it has 1 virtual machine for constraint method.The bidder's specification for auction is given below in TableVI.

Table VI: CONSTRAINT Table With Price Per License By Bidders For Slot 3

SLOTS/BIDDERS	VM REQUIRED	PRICE PER VM
C1	2	30
C2	1	28
C3	1	35
C4	1	10

Based on the Auctioning the virtual machines algorithm, the selected bidder is C3 and cost acquired in auction is 35.

- For slot 4, it has 10 virtual machines. 8 virtual machines are given for normal auction and 2 remaining virtual machine is given to constraint method. The bidder's specification for auction is given below in Table VII.

Table VII: Auction Table With Price Per VM By Bidders For Slot4

SLOTS/BIDDERS	VM REQUIRED	PRICE PER VM
B1	7	20
B2	5	30
B3	8	10
B4	5	35
B5	3	30

Table VIII: Auction Table For Selected Bidder With Price Per VM For Slot4

SLOTS/BIDDERS	VM REQUIRED	PRICE PER VM
B4	5	35
B5	3	30

Based on the Auctioning the virtual machines algorithm, the selected bidder are B4 and B5.The cost acquired in auction is 265.

- For slot 4, it has 2 virtual machine for constraint method.The bidder's specification for auction is given below in TableIX.

Table IX: CONSTRAINT Table With Price Per License By Bidders For Slot 4

SLOTS/BIDDERS	VM REQUIRED	PRICE PER VM
C1	2	35
C2	1	28
C3	1	25
C4	1	20

- The selected bidders from constraint table is C1 and the cost acquired is 70.

Algorithm 3 Auction algorithm using knapsack

1. **Procedure** Auction(UL,S,B)
2. **foreach** slots unused virtual machines are allocated
3. Set λ_m=80% of unused virtual machines (UL) and τ_m =20% of unused virtual machines (UL)
4. Set cost=0
5. Select the bidders according to the bidder selection strategies
6. Set W= λ
7. **for** j=0 to W
8. B[0,j]=0
9. **End for**
10. **For** i=1 to n
11. B[i,0]=0
12. **End for**
13. **for** i=1 to n
14. **for** j=0 to W
15. **if** $w_i <= j$ //then item i is selected for auction
16. **if** $v_i + B[i-1, j-w_i] > B[i-1,j]$
17. $B[i,j] = v_i + B[i-1,j-w_i]$
18. **else**
19. B[i,j] = B[i-1,j]
20. **End if**
21. **else**
22. B[i,j] = B[i-1,j]
23. **End if**
24. **End for**
25. **End for**
26. set i=n, k=W
27. **while** i,k > 0
28. **if** B[i,k] =B[i-1, k]-

29. Select the i^{th} item and update cost
30. set i= i-1 and k=k-w[i]
31. **else**
32. i=i-1
33. **End if**
34. **End while**
35. set W= τ
36. **if** there is constraint **then**
37. **if** cost > worth of virtual machines **then**
38. **if** virtual machines required ≤ W
39. update cost
40. **end if**
41. **Else**
42. Set λ= τ, **goto** step 5
43. **end if**
44. **end foreach**
45. **return** cost
46. **end procedure**

Revenue Calculation Algorithm: The fourth algorithm namely, Revenue Calculation, combines all the above mentioned three algorithms and compute cost by satisfying all tenants and bidders. The maximized profit is the sum of the cost generated by algorithm 2 and algorithm 3.

Algorithm 4 Revenue Calculation

1. **Procedure revenue**(SLA)
2. //call Algorithm 1
3. Get min_lic // the needed minimum virtual machine
4. Set L=min_vm
5. //call Algorithm 2
6. Get cost and L_k^{rem}
7. **if** L_k^{rem} > 0 **then**
8. choose slot which has unused virtual machine
9. //call Algorithm 3
10. Get c[i] // auctioned cost
11. max_profit=cost + au_cost
12. **end if**
13. **return** max_profit
14. **end procedure**

Revenue Calculation:

Slots	Tenant Profit in Rupees	Auction Profit in Rupees	Total Profit in Rupees
Slot 1	1280	-	1280
Slot 2	1060	-	1060
Slot 3	840	125	965
Slot 4	700	335	1035
Total (In Rupees)	3880	460	4340

The total profit gained from tenant maximization is Rs.3,880 and when auction method is used the profit increased is Rs.460.Therefore,the total profit obtained after using auctioning is Rs.4340 and the total profit percentage increased is 18.4%.

Evaluation: This system is evaluated successfully in the Java (JDK 1.7.0_45). By implementing these algorithms for the above specified SLA, the virtual machines needed for each slot is 20, 20, 16 and 8 respectively. Hence, the minimum virtual machines required to satisfy all the tenants is 20. After on-boarding tenants, based on the user requirements, the unused virtual machines available in each slot is 0, 0, 6 and 10. The revenue generated by satisfying all the tenants is 3880. After implementing auction method, the revenue generated is 460. Therefore the total revenue generated by these algorithms is 4340.

To reduce the wastage of unused virtual machines in slots is given to auction. In auctioning method, to get optimum results knapsack algorithm is implemented. It results in higher profit to the service provider.

RESULTS

This section details the results acquired by Java (JDK 1.7.0_45). The increase in the revenue with auction and without auction for above specified SLA is demonstrated with the help of Fig. 2.

Fig. 2: Number of Clients Vs Revenue Graph with and without Auction.

The revenue generated in the proposed system is higher than the revenue generated in the existing system. This increase in the revenue is demonstrated with the help of Fig. 3.

Fig. 3: Number of Clients Vs Revenue Graph, by Proposed System and by Existing system.

In the proposed algorithm, even if there is licenses remained unused after auction. It is given for constraint method. By doing this, wastage of resources is reduced as well as revenue also increased for the service provider.

CONCLUSION

This paper deals with the resource allocation based on dynamic SLA specification by Clients and auctioning. Here, resources are allocated without violating the SLA, by pre-determining the resources required and job completion percentage specified by the clients in earlier, which helps in reducing SLA violation. Therefore, Service providers are not required to pay penalty and improves their QoS. To reduce the wastage of unused virtual machines in slots is given to auction. In auctioning method, to get optimum results dynamic programming 0-1 Knapsack problem algorithm is implemented. It results in higher profit to the service provider.

REFERENCES

1. Bipin B. Nandi, Ansuman Banerjee, Sasthi C. Ghosh, Nilanjan Banerjee, 2013. Dynamic SLA Based Elastic Cloud Service Management: A SaaS Perspective, IFIP/IEEE International Symposium on Integrated Network Management (IM2013).
2. Massimiliano Rak and Antonio Cuomo, 2013. A Proposal of a Simulation-based Approach for Service Level Agreement in Cloud, 27th International Conference on Advanced Information Networking and Applications Workshops.
3. Duo Liu, Utkarsh Kanabar and Chung-Horng Lung, 2013. A light Weight SLA Management Infrastructure for Cloud Computing, 2013 26th IEEE Canadian Conference of Electrical and Computer Engineering (CCECE).
4. Bertsekas, D., 1992. Auction algorithms for network flow problems: A tutorial introduction, Computat. Optimiz. Appl., 1(1): 7-66.
5. Oshri Naparstek, 2014. Fully Distributed Optimal Channel Assignment for Open Spectrum Access, IEEE Transactions on signal processing, 62(2).