IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 5, May 2018

Distributed Load Balancing and Scheduling Algorithm for Resource Allocation in Cloud

AR. Meenakshi[1], R. Divya Sopna[2]

ME Scholar, Department of Computer Science and Engineering and Technology,
Sri Raaja Raajan College of Engineering and Technology, Amaravthi Pudur, Tamil Nadu, India [1]

Assistant Professor, Department of Computer Science and Engineering and Technology,
Sri Raaja Raajan College of Engineering and Technology, Amaravthi Pudur, Tamil Nadu, India [2]

Abstract: Cloud computing is the latest distributed computing paradigm and it offers tremendous opportunities to solve large scale scientific problems. However, it presents various challenges that need to be addressed in order to be efficiently utilized for workflow applications. Although the workflow scheduling problem has been widely studied, there are very few initiatives tailored for Cloud environments. Furthermore, the existing works fail to either meet the user's Quality of Service (QoS) requirements or to incorporate some basic principles of Cloud computing such as the elasticity and heterogeneity of the computing resources. This work proposes a resource provisioning and scheduling strategy for scientific workflows on Infrastructure as a Service (IaaS) Clouds. We present an algorithm based on the meta-heuristic optimization technique, Particle Swarm Optimization (PSO), which aims to minimize the overall workflow execution cost while meeting deadline constraints. Our heuristic is evaluated using CloudSim and various well-known scientific workflows of different sizes. The results show that our approach performs better than the current state-of-the-art algorithms.

Keywords: QoS, PSO, SLAs, IaaS, PaaS, SLA, SAAS, GAP.

I. INTRODUCTION

Cloud computing is the delivery of computing as a service rather than a product, whereby shared resources, software and information are provided to users over the network. Cloud computing providers deliver application via the Internet, which are accessed from web browser, while the business software and data are stored on servers at a remote location.
Cloud computing is a new and emerging trends in distributing computing that facilitate software application platforms, and hardware infrastructures as a service. Cloud service provider offers these services based on customized Service Level Agreements (SLAs), which defined user's required Quality of Service (QoS) parameters. Cloud computing reduces investments on various resource like hardware and software resource allows to be leased and released. It reduces initial investment, maintenance costs and operating cost. Cloud services are hosted on service provider's own infrastructures or on third parties cloud infrastructure providers. Mainly three kinds of services are delivered; Platform as a Service (PaaS), Infrastructure as a Service (IaaS) and Software as a Services (SaaS). Cloud users using this service, whenever needed to a according to their demands using pay-per-use models. Clouds provided the ability to adjust resources capacity according to the changing demands of the applications, often called auto scaling. However, giving users more controls also required the developments of new method for task scheduling and resource provisioning. Resource management decision is required to cloud scenarios not only have to take into account performance related metrics such as workflows make spam or resource utilizations, but must also considers budget constraints, since the resources from commercial clouds, Usually have a monetary costs associated with them. To gain insight to resource a management challenges, when executing a scientific workflow ensembles on clouds. We address a new and important problems of maximizing the numbers of completed workflows from an ensembles a under both budget and deadline constraint.
Besides many applications, cloud computing environment can be used for workflow execution also. Execution of a workflow involves workflow scheduling. Workflow scheduling involves mapping of workflow tasks with available resources in such a way that some predefined criteria is met. Workflow scheduling is well known NP-complete problem [4] and key issue in workflow management system. Moving workflows to Cloud computing enable us to exploit the benefits of cloud for workflow execution. Scheduling can be multi objective also. The multi objective nature of scheduling is more difficult to solve. Many heuristic and meta-heuristic approaches have been proposed by different researchers for workflow scheduling. At present, workflow scheduling algorithms for cloud systems focus on two major parameters viz. cost and time. As cloud uses pay-as-you-go model, all services incur cost. Cost mainly dependents on QoS (Quality of Service) offered. Service providers charge higher for higher QoS and lower for lower QoS. Early and reliable execution of jobs is another important factor from cloud user's point of view, but it incurs more

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 5, May 2018

cost. Users may require earlier reliable completion of their workflow tasks within manageable cost along with other QoS requirements. These kinds of requirements make workflow scheduling on clouds more important and complex.

II. LITERATURE REVIEW

Ranjit Singh et al In this paper we consider deadline as the major constraint and propose a score based deadline constrained workflow scheduling algorithm that executes workflow within manageable cost while meeting user defined deadline constraint. The algorithm uses the concept of score which represents the capabilities of hardware resources. This score value is used while allocating resources to various tasks of workflow application. The algorithm allocates those resources to workflow application which are reliable and reduce the execution cost and complete the workflow application within user specified deadline. [1]

S.Mohana priya et al we develop the concepts of "dynamic data flows" which utilize alternating tasks as additional control over the data flows cost and QoS. Further, we formalize an optimization problem to representing deployment and runtime resources provisioning that allows us to balance the application's QoS, value and the resource costs. We proposed two greedy heuristics, centralized and shared, based on the variable sized been packing algorithm and compare against a Genetic Algorithm (GA) based heuristics that gives a near optimal solution. A large scale simulation study, using the linear roads benchmark and VM performances trace from the AWS public cloud, shows that while GA based heuristic provides a better quality schedule, the greedy heuristics are more practical, and can intelligently utilize cloud elasticity to mitigate the effect of variability, both in input data rates and cloud resource performance, to meet the QoS of fast data applications. [2]

Anterpreet Kaur et al A cloud workflow system is a type of platform service which facilitates the automation of distributed applications based on the novel cloud infrastructure. Many scheduling policies have been proposed till now which aim to maximize the amount of work completed while meeting QoS constraints such as deadline and budget. However many of them are not optimal to incorporate some basic principles of Cloud Computing such as the elasticity and heterogeneity of the computing resources. [3]

J. M. Wilson et al The generalized assignment problem (GAP), the 0–1 integer programming (IP) problem of assigning a set of n items to a set of m knapsacks, where each item must be assigned to exactly one knapsack and there are constraints on the availability of resources for item assignment, has been further generalized recently to include cases where items may be shared by a pair of adjacent knapsacks. This problem is termed the generalized assignment problem with special ordered sets of type 2 (GAPS2) [4].

M. Qiu et al each varied execution time as a probabilistic random variable and solves heterogeneous assignment with probability(HAP) problem. The solution of the HAP problem assigns a proper FU type to each task such that the total cost is minimized while the timing constraint is satisfied with a guaranteed confidence probability. A probabilistic approach to high-level synthesis of special-purpose architectures for real-time embedded systems using heterogeneous functional units with probabilistic execution times. For the heterogeneous assignment with probability(HAP) problem, algorithms Path Assign and Tree Assign, were proposed to give optimal solutions when the input graphs are a simple path and a tree, respectively. Two other algorithms, one is optimal and the other is near-optimal heuristic, were proposed to solve the general problem. [5].

III. PREVIOUS STUDIES

In recent years ad-hoc parallel data processing has emerged to be one of the killer applications for Infrastructure-as-a-Service (IaaS) clouds. Major Cloud computing companies have started to integrate frameworks for parallel data processing in their product portfolio, making it easy for customers to access these services and to deploy their programs. However, the processing frameworks which are currently used have been designed for static, homogeneous cluster setups and disregard the particular nature of a cloud. Consequently, the allocated compute resources may be inadequate for big parts of the submitted job and unnecessarily increase processing time and cost. To reduce the impact of performance variation of public Cloud resources in the deadlines of workflows, we proposed a new algorithm, called EIPR, which takes into consideration the behavior of Cloud resources during the scheduling process and applies replication of tasks to increase the chance of meeting application deadlines.A.EIPR Algorithm The existing algorithm performs following steps

1. Combined Provisioning and Scheduling
2. Data-Transfer Aware Provisioning Adjust
3. Task Replication

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 5, May 2018

Combined Provisioning and Scheduling The first step of the EIPR algorithm consists in the determination of the number and type of VMs to be used for workflow execution as well as start and finish time of each VM (provisioning) and the determination of ordering and placement of tasks on such allocated resources (scheduling). The provisioning and scheduling problems are closely related, because the availability of VMs affects the scheduling, and the scheduling affects finish time of virtual VMs. Therefore, a more efficient scheduling and provisioning can be achieved if both problems are solved as one rather than independently.

A. Demerits of Existing Work

- A policy issue remains as how to decide the mapping adaptively so that the resource demands of VMs are met while the number of PMs used is minimized.
- This is challenging when the resource needs of VMs are heterogeneous due to the diverse set of applications they run and vary with time as the workloads grow and shrink. The two main disadvantages are overload avoidance and green computing.
- Expensive
- Complex
- Increases data base organization
- The processing framework then takes care of distributing the program among the available nodes and executes each instance of the program on the appropriate fragment of data. Most notably, Nephele is the first data processing framework to include the possibility of dynamically allocating/ deallocating different compute resources from a cloud in its scheduling and during job execution.

IV. PROPOSED WORK

A. Proposed work

The proposed priority algorithm helps cloud admin to decide priority among the users and allocate resources efficiently according to priority. This resource allocation technique is more efficient than grid and utility computing because in those systems there is no priority among the user request and cloud administrator is randomly taking decision and he is giving priority to those user who have submitted their job first that is based on first come first serve method. But with the advent of cloud computing and by using this implemented priority algorithm, the cloud admin can easily take decision based on different parameters discussed earlier to decide priority among different user request so that admin can efficiently allocate the available resources and with cost-effectiveness as well as satisfaction from users. Once the user's request will be received at the cloud end, after that according to the user's requirement, the resources will be checked for assigning to the user. Batches of the user's requirement will be created according to the type of task, the amount of processor required by the user, and time for the execution of the user. If the resources are not available then the user needs to wait for the resources to be available. The user's waiting request will be compared with all the waiting resources and priority will be assigned accordingly. The throughput value is calculated according to the usage of the processor and ram. If the request of two same requirements having the same priority then at that point of time the resources will be allocated on the basis of FCFS(First Come First Served).

B. Advantages of proposed system

The advantages of proposed systems are as follows:

- Dynamic resource allocation
- Parallelism is implemented
- Designed to run data analysis jobs on a large amount of data
- Many Task Computing (MTC) has been developed
- Less expensive
- More Effective
- More Faster
- In this work, present the design and implementation of an automated resource management system that achieves a good balance between the two goals. Two goals are **overload avoidance** and **green computing**.
- **Overload avoidance**: The capacity of a PM should be sufficient to satisfy the resource needs of all VMs running on it. Otherwise, the PM is overloaded and can lead to degraded performance of its VMs.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 5, May 2018

- **Green computing**: The number of PMs used should be minimized as long as they can still satisfy the needs of all VMs. Idle PMs can be turned off to save energy.

V. METHODOLOGY

A. Network Simulator

Network Simulator is a name for series of discrete event network simulators, specifically NS-1, NS-2 and NS-3. All of them are discrete-event network simulators, primarily used in research and teaching. NS-3 is free software, publicly available under the GNU GPLv2 license for research, development, and use. The goal of the ns-2 project is to create an open simulation environment for networking research that will be preferred inside the research community. It should be aligned with the simulation needs of modern networking research. It should encourage community contribution, peer review, and validation of the software. Since the process of creation of a network simulator that contains a sufficient number of high-quality validated, tested and actively maintained models requires a lot of work, ns-2 project spreads this workload over a large community of users and developers. Presently, ns-2 consists of over 300,000 lines of source code, and there is probably a comparable amount of contributed code that is not integrated directly into the main distribution (many forks of ns-2 exist, both maintained and unmaintained). It runs on GNU/Linux, FreeBSD, Solaris, Mac OS X and Windows versions that support Cygwin. It is licensed for use under version 2 of the GNU General Public License. Though NS-3 is actively developed this is not compactable for the work done in NS-2. So for this project NS-2 is highly supportive.

B. CYGWIN

Cygwin is a Unix-like environment and command-line interface for Microsoft Windows. Cygwin provides native integration of Windows-based applications, data, and other system resources with applications, software tools, and data of the Unix-like environment. Thus it is possible to launch Windows applications from the Cygwin environment, as well as to use Cygwin tools and applications within the Windows operating context.

It consists of two parts: a dynamic-link library (DLL) as an API compatibility layer providing a substantial part of the POSIX API functionality, and an extensive collection of software tools and applications that provide a Unix-like look and feel.Cygwin was originally developed by Cygnus Solutions, which was later acquired by Red Hat. It is free and open source software, released under the GNU General Public License version 3. Today it is maintained by employees of Red Hat, NetApp and many other volunteers.

VI. SCHEDULING ALGORITHM

In proposed priority based scheduling algorithm we have modified the scheduling heuristic for executing highest priority task with advance reservation by preempting best-effort task as done. Algorithm shows the pseudo codes of priority based scheduling algorithm (PBSA). Scheduling is presented which helps in achieving Service Level Agreement with quick response from the service provider. In our proposed approach Quality of Service metric such as response time is achieved by executing the high priority jobs (deadline based jobs) first by estimating job completion time and the priority jobs are spawned from the remaining job with the help of Task Scheduler. Scheduling and proposed a particle swarm optimization (PSO) algorithm which is based on small position value rule. In order to improve the efficiency the optimizing task scheduling is necessary. In cloud computing resources distribute all over the world, and the data usually is bigger and the bandwidth often is narrower, these problems are more important. In this paper, the author presented the task scheduling optimizing method in cloud computing, and formulates a model for task scheduling to minimize the cost of the problem and solved it by a PSO algorithm. Experimental result manifests that the PSO algorithm both gains optimal solution and converges faster in large tasks than the other two. Moreover, running time is shorter than the other two too and it is obvious that PSO is more suitable to cloud computing.

A. Priority based Job Scheduling Algorithm in Cloud Computing

Proposed a new job scheduling algorithm in cloud computing by using mathematical statistics. This algorithm made its foundation on the priority property that why it is known as Priority-Based Algorithm. It is based on multiple criteria decision making model. In 1980 Thomas Saaty was first on to develop a model that make pair wise comparison based on multiple criteria and multiple attributes and named it as Analytical Hierarchy Process (AHP). AHP is purely based on Consistent Comparison Matrix, so by making the use of AHP, comparison matrices are computed according to the attributes and criteria's accessibilities. In this algorithm, each job requests a resource which has a pre-determined priority. So according to resources accessibilities, comparison matrices of each jobs is computed. Author also computes the comparison matrix of resources which will help later for jobs picking. Then author compute priority vectors (vector of weights) for each the comparison matrix and finally a normal matrix of all jobs is computed named as Δ. Similarly, normal matrix of all resources is computed and marks this matrix as γ. The next step of the algorithm is to compute

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 5, May 2018

Priority Vector of S (PVS), where S is set of jobs. PVS is calculated by multiplying matrix Δ with matrix γ. At final step, algorithm chooses the job with maximum calculated priority on basis of that suitable resource is allocated to that job. Now the list of jobs is updated and the scheduling process continues till all the jobs are assigned to suitable resource. Experimental results indicate that the algorithm has reasonable complexity. But there some issues such as complexity, consistency and finish time.

ALGORITHM: Priority Based Scheduling Algorithm (PBSA)
1. **Input:** UserServiceRequest
2. //call Algorithm 2 to form the list of task based on priorities
3. get globalAvailableVMList and gloableUsedVMList and also available ResourceList from each cloud schedular
4. // find the appropriate VMList fromeach cloud scheduler
5. **if** AP(R,AR) != ϕ **then**
6. // call the algorithm 1 load balancer
7. deployableVm=load-balancer(AP(R,AR))
8. Deploy service on deployableVM
9. deploy=true
10. **Else if** R has advance reservation and best-effort task is running on any cloud **then**
11. // Call algorithm 3 CMMS for executing R
with advance reservation
12. Deployed=true
13. **Else if** globalResourceAbleToHostExtraVM **then**
14. Start newVMInstance
15. Add VMToAvailbaleVMList
16. Deploy service on newVM
17. Deployed=true
18. **Else**
19. queue serviceReuest until
20. queueTime > waitingTime
21. Deployed=false
22. **End if**
23. If deployed then
24. return successful
25. terminate
26. **Else**
27. return failure
28. Terminate

B. Cloud min-min scheduling (CMMS)
Min-min scheduling is popular greedy algorithm. The dependences among tasks not careful in original min min algorithm. Thus in the dynamic min-min algorithm used , authors uphold the task dependences by updating the map able task set in every preparation step. The tasks whose precursor tasks are all assigned are placed in the map able task set. Algorithm 3 shows the quasi codes of the CMMS algorithm. A cloud scheduler record implementation schedule of all resources using a slot. Once an AR task is assigned to a cloud, first reserve availability in this cloud will be checked by cloud scheduler. Then best-effort task can be pre-empted by AR task, the only case once most of resources are earmarked by some other AR task. Later there are not enough resources left for this AR task in the obligatory time slot. If the AR task is not disallowed, which means there are enough resources obtainable for the task, a set of required VMs are selected randomly. The estimated finish time of task may not be same as real finish time due to the resource argument within individual cloud. Later to adjust the resource allocation animatedly based on the latest available information writers propose an online adaptive scheduling process. In future online adaptive procedure the remaining static resource distribution will be re-evaluate recurrently with a predefined incidence. In each reevaluation, the schedulers will re-calculate the projected finish time of their tasks. Note that a scheduler of a assumed cloud will only reconsider the tasks that are in the jobs succumbed to this cloud, not the errands that are assigned to this cloud.

Algorithm 3 Cloud min-min scheduling (CMMS)
Require: A set of tasks, m different clouds ETM matrix
Ensure: A schedule generated by CMMS
1. For a mappable task set P
2. **While** there are tasks not assigned **do**

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 5, May 2018

3. Update mappable task set P
4. **For** I = task vi ∈ P **do**
5. Send task check requests of vi to all other cloud schedulers
6. Receive the earliest resource available time response and And list of task with their priorities form all other cloud scheduler
7. Find the cloud Cmin(vi) giving the earliest finish time of vi, assuming no other task preempts vi
8. **End for**
9. Find the task-cloud pair (vk, Cmin(vk)) with earliest finish time in the pairs generated in forloop
10. Assign task v k to cloud Dmin(vk)
11. Remove v k form P
12. Update the mappbale task set P
13. **End while**

Algorithm: To compute and assign the priority for each request based on the threshold value and allocate the service to each request‘s.
Step 1: [Read the clients request data i.e, time, importance, price, node and requested server name] Insert all values into the linked list
Step 2: [For each request and its tasks find the **time** priority value based on the predefined conditions] Assign priority value to each task for the client‘s request.

t_p[i] = priority value

Step 3: [For each request and its tasks find the **node** priority value based on the predefined conditions]
Assign priority value to each task for the client‘s request.

n_p[i] = priority value;

Step 4: [For each client‘s input data check whether it is within the threshold value or not]
if (input value is within the threshold limit and total node <= available node)
[Add respective computed time and node priority value and other parameters like importance and price]
Sum[k] = t_p[i] + n_p[i] + importance + price
Print ―Ready to execute available node = available node – total node
else if (input value is within the threshold limit)
sum[k] = t_p[i] + n_p[i] + importance + price
print ―within the limit but it is in queue
else
print ―Exeed the condition
Step 5: [Sort the sum[k] values]
Step 6: Client‘s request is ready to execute from least values of sum[k]
Stop

In order to run particular model huge computational resources such as server, memory in terms of storage disk, processors, software etc are needed. Also some jobs are to be executed in parallel and some others in sequential manner. In that situation job type is also very important parameter.
In a cloud environment type of user that is whether the user is internal to a cloud (in case of private cloud) or he is external to cloud(in case of public cloud) is also another important parameter to be considered during job submission. So the developed priority algorithm discusses in detail how efficiently it will help cloud admin to decide or calculate priority among the user requests.
After the successful execution of resource allocation algorithm, the jobs requested by users needs to be submitted. The main difficulties in the resource allocation in a cloud system are to take proper decision for job scheduling, execution of job, managing the status of job etc. Apart from traditional best fit and bin packing algorithm in this paper an algorithm is developed for the job allocation in the cloud environment to be decided by the cloud administrator. priority based on the client and server requirements and requests by the users. In the present algorithm to decide the resource allocation in a better and impartial way, a technique based on threshold of all the parameters (both client and server side) is considered. For example the requested number of processors cannot be more than 20 etc. (server) and a job maximum run time will be 200 hrs (user).

VII. RESULTS AND DISCUSSION

A. Setup
We evaluate our performance based on the priority by simulating scheduling algorithm. One by one Simulation of the working group completed in 10 games. In each execution Simulation, a group of 70 different analog service Applications (ie jobs), and each includes a service request up to 18 subtasks. We believe in Simulation of clouds. All 70

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 5, May 2018

will be subject to random cloud service requests any time soon. In these requests services 70, 15 Application is in AR mode, while the rest is the best way to work with different SLA objectives. That Table 1 Parameter set randomly in simulation According to their maximum and minimum values. Since we only focus on the planning algorithms, we do simulations locally without implementing in any exiting cloud system or using VM interface API.

TABLE I
PARAMETERS AND ITS RANGE

Parameter	Minimum	Maximum
No. of virtual machine in cloud	23	120
No. of CPU in a VM	1	7
Disk Space	8000	100000
Memory	16	2048
Speed	100	1000

B. Result

In Fig 1 shows the average execution job loose situation. We realized that the algorithm PBSA. The minimum average execution time. Resource Parameters when work occurs AR work best be replaced by. Such as Resource contention at least loosened, it is expected that Target part-time work is nearing completion of the actual time. Therefore Adaptation procedure does not affect the date of execution significant.

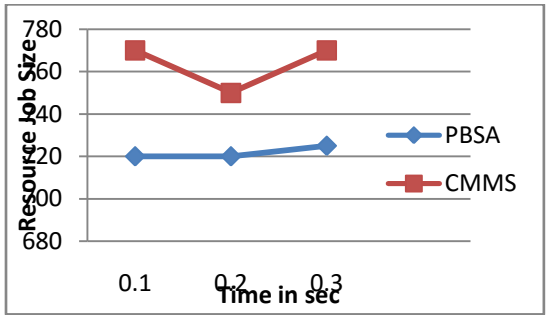
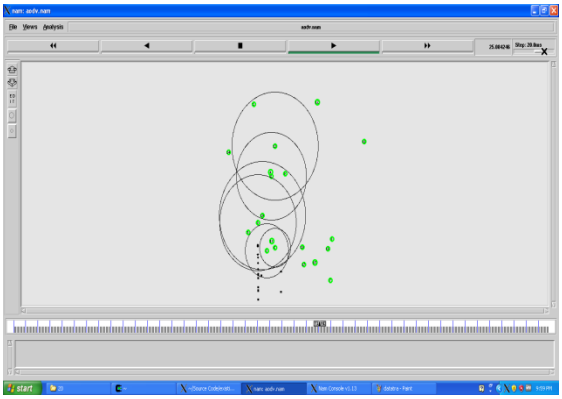


Fig 1. Average job execution time in loose situation



Under prove difficult situation shown in Fig 2 PBSA behavior CMMS better. In stressful situations the scramble for resources more so when the work actually completed it is often later than expected arrival. Because AR preemption works the best, the process of adaptation and upgrade Information more meaningful works in a difficult situation.

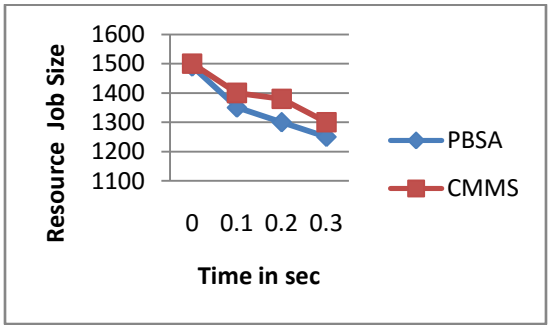
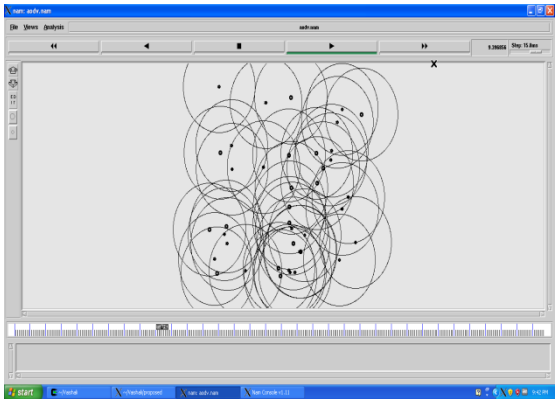


Fig 2. Average job execution time in tight situation

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 5, May 2018

This requires dynamically estimating the energy consumed by methods during execution. This take inspiration from the recent PowerTutor model which accounts for power consumption of CPU, LCD screen, GPS, WiFi, 3G, and audio interfaces on HTC Dream and HTC Magic phones. Power Tutor indicates that the variation of estimated power on different types of phones is very high, and presents a detailed model for the HTC Dream phone which is used in our experiments. We modify the original PowerTutor model to accommodate the fact that certain components such as GPS and audio have to operate locally and cannot be migrated to the cloud.



By measuring the power consumption of the phone under different cross products of the extreme power states, PowerTutor model further indicates that the maximum error is 6.27% if the individual components are measured independently. This suggests that the sum of independent component-specific power estimates is sufficient to estimate overall system power consumption. Using this approach we devise a method with only minor deviations from the results obtained by PowerTutor. We implement this energy estimation model inside the ThinkAir Energy Profiler and use it to dynamically estimate the energy consumption of each running method.

VIII. CONCLUSION AND FUTURE WORK

Cloud computing resources means that in the selection, implementation and management time, management software (e.g., database server, load Balancers, etc.) and hardware resources (for example, CPU Storage, networking, etc.), in order to ensure security application performance. These techniques to improve response time, performance, save Energy, quality of service, SLA. The ultimate goal of resources Configuration is to maximize the benefits of the cloud Prospects for service providers and cloud the user's perspective, in order to reduce costs. There are many current challenges Strategic resource allocation. A mechanism to overcome the challenges faced by the prior art It must be used. The architecture must be proposed it is suitable for data-intensive applications and high performance computing Also on the actual workload. Mechanism must It recommends that effective use of cloud computing resources to enable QoS and SLA violations in meeting minimization Dynamic Allocation of clouds. Also These mechanisms should also be used to conFig SaaS and IaaS users.

REFERENCES

[1] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, I. Brandic, “Cloud computing andemerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility”, Future Generation ComputerSystems 25 (2009) 599–616.
[2] S. Pandey, L. Wu, S. M. Guru, and R. Buyya, “A particle swarm optimization-based heuristic for scheduling work flow applicationsin cloud computing environments,” in AINA ’10: Proceedings of the 2010, 24th IEEE International Conference on Advanced Information Networking and Applications, pages 400–407, Washington, DC, USA, 2010, IEEE Computer Society.
[3] M. Salehi and R. Buyya, “Adapting market-oriented scheduling policies for cloud computing,”I n Algorithms and Architectures for Parallel Processing, volume 6081 of Lecture Notes in Computer Science, pages 351–362.Springer Berlin / Heidelberg, 2010.
[4] J. M. Wilson, “An algorithm for the generalized assignment problemwith special ordered sets,” Journal of Heuristics, 11(4):337–350, 2005.
[5] M. Qiu and E. Sha, “Cost minimization while satisfying hard/soft timing constraints for heterogeneous embedded systems,” ACM Transactions on Design Automation of Electronic Systems (TODAES), vol. 14, no. 2, pp. 1–30, 2009.
[6] M. Qiu, M. Guo, M. Liu, C. J. Xue, and E. H.-M. S. L. T. Yang, “Loop scheduling and bank type assignment for heterogeneous multibank memory,” Journal of Parallel and Distributed Computing(JPDC), vol. 69, no. 6, pp. 546–558, 2009.
[7] A. Dogan and F. Ozguner,“Matching and scheduling algorithms for minimizing execution time and failure probability of applications in heterogeneous computing,” IEEE Transactions on Parallel and Distributed Systems, pp. 308–323, 2002.
[8] T. Hagras and J. Janecek, “A high performance, low complexity algorithm for compile-time task scheduling in heterogeneous systems,” Parallel Computing,vol. 31, no. 7, pp. 653–670, 2005.
[9] “Adaptive Management of Virtualized Resources in Cloud Computing Using Feedback Control,” in First International Conference on Information Science and Engineering, April 2010, pp. 99-102.
[10] W. E. Walsh, G. Tesauro, J. O. Kephart, and R. Das, “Utility Functions in Autonomic Systems,” in ICAC’04: Proceedings of the First International Conference on Autonomic Computing. IEEE Computer Society, pp. 70–77, 2004.

[11] Jiayin Li, Meikang Qiu, Jian-Wei Niu, Yu Chen, Zhong Ming, "Adaptive Resource Allocation for Preempt able Jobs in Cloud Systems," in 10th International Conference on Intelligent System Design and Application, Jan. 2011, pp. 31-36.
[12] Yazir Y.O., Matthews C., Farahbod R., Neville S., Guitouni A., Ganti S., Coady Y., "Dynamic resourceal location based on distributed multiple criteria decisions in computing cloud," in 3rd International Conference on Cloud Computing, Aug. 2010, pp. 91-98.
[13] Goudarzi H., Pedram M., "Multi-dimensional SLA-based Resource Allocation for Multi-tier Cloud Computing Systems,"in IEEE International Conference on Cloud Computing, Sep. 2011, pp. 324- 331.
[14] Shi J.Y., Taifi M., Khreishah A.,"Resource Planning for Parallel Processing in the Cloud," in IEEE 13th International Conferenceon High Performance and Computing, Nov. 2011, pp. 828-833.
[15] Aoun R., Doumith E.A., Gagnaire M., "Resource Provisioning for Enriched Services in Cloud Environment," IEEE Second International Conference on Cloud Computing Technology and Science, Feb. 2011, pp. 296-303.
[16] T. Erl, "Service-oriented Architecture: Concepts, Technology, and Design", Upper Saddle River, Prentice Hall, 2005.
[17] F. Chong, G. Carraro, and R. Wolter,"Multi-Tenant Data Architecture", Microsoft Corporation, 2006.
[18] E. Knorr, "Software as a service: The next big thing", InfoWorld, March 2006.
[19] PLAStiCC: Predictive Look-Ahead Scheduling for Continuous dataflows on Clouds by Alok Kumbhare, Yogesh Simmhan and Viktor K. Prasanna.
[20] Efficient Parallel Data Processing in the Cloud by Thanapal.P, Nishanthi.S.P
[21] Sheng Di and Cho-Li Wang," Dynamic Optimization of Multi-Attribute Resource Allocation in Self-Organizing Clouds", IEEE Transactions on parallel and distributed systems, - 2013.
[22] V.Vinothina, Dr.R.Sridaran, Dr.padmavathiganapathi," A Survey on Resource Allocation Strategies in Cloud Computing "International Journal of Advanced Computer Science and Applications, Vol. 3, No.6, 2012.
[23] Qi Zhang, Eren G¨urses, Raouf Boutaba, Jin Xiao," Dynamic Resource Allocation for Spot Markets in Clouds", Journal of computer science-2012.
[24] Zhen Xiao, Senior Member, IEEE, Weijia Song, and Qi Chen," Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment", IEEE Transactions on parallel and distributed systems, vol. 24, no. 6, June 2013.
[25] Ts`epomofolo, R Suchithra," Heuristic Based Resource Allocation Using Virtual Machine Migration: A Cloud Computing Perspective", International Refereed Journal of Engineering and Science (IRJES) Volume 2, Issue 5(May 2013)
[26] S. Callaghan, P. Maechling, P. Small, K. Milner, G. Juve, T. Jordan, E. Deelman, G. Mehta, K. Vahi, D. Gunter, K. Beattie, and C. X. Brooks, "Metrics for heterogeneous scientific workflows: A case study of an earthquake science application," International Journal of High Performance Computing Applications, vol. 25, 2011.
[27] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, "The cost of doing science on the cloud: The montage example," in 2008 ACM/IEEE Conference on Supercomputing (SC 08), 2008.
[28] J. V¨ockler, G. Juve, E. Deelman, M. Rynge, and G. B. Berriman, "Experiences using cloud computing for a scientific workflow application," in 2nd Workshop on Scientific Cloud Computing 2011.
[29] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "A performance analysis of EC2 cloud computing services for scientific computing," in Cloud Computing, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, O. Akan et al., Eds. Springer Berlin, 2010.
[30] K. Keahey, M. Tsugawa, A. Matsunaga, and J. Fortes, "Sky computing," IEEE Internet Computing, vol. 13, no. 5, 2009.
[31] D. Durkee, "Why cloud computing will never be free," Communications of the ACM, vol. 53, no. 5, May 2010.
[32] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience, vol. 41, no. 1, Jan. 2011.
[33] L. Fernando, B. Edmundo, M. Madeira M, "HCOC: A Cost Optimization Algorithm for Workflow Scheduling in Hybrid Clouds", Journal of Internet Services and Applications, Springer,2011.
[34] P. Marshall, K. Keahey, and T. Freeman, "Elastic site: Using clouds to elastically extend site resources," in 10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2010), 2010.
[35] H. Kim, Y. el-Khamra, I. Rodero, S. Jha, and M. Parashar, "Autonomic management of application workflows on hybrid computing infrastructure," Scientific Programming, vol. 19, April 2011.
[36] J. Yu, R. Buyya, and C. Tham, "Cost-Based scheduling of scientific workflow application on utility grids," in First International Conference on e-Science and Grid Computing, 2005.
[37] S. Abrishami, M. Naghibzadeh, and D. Epema, "Cost-driven scheduling of grid workflows using partial critical paths," in 11th IEEE/ACM International Conference on Grid Computing, 2010