Journal of Theoretical and Applied Information Technology
31st March 2016. Vol.85. No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

K-TIER SEPARATION BASED ABSTRACTION REFINEMENT SCHEDULERS FOR PARALLEL JOB IN MULTIPLE CLOUD CENTERS

MR.C.ANTONY, DR.C.CHANDRASEKAR, DR.S.NITHYA REKHA
Research Scholar, Bharathiyar University, Coimbatore-641 046, Tamilnadu, India.
Professor, Department of Computer Science, Periyar University, Salem-636011, Tamilnadu, India
Assistant Professor, Dept. of Computer Science and Engineering, Qassim Private Colleges
Al-Qassim, Buraidah,Kingdom of Saudi Arabia
E-mail:ananant26@gmail.com, ccsekar@gmail.com

ABSTRACT

Current approaches to enforce cloud computing for complex applications, processed in remote data centers are based on parallel processing capabilities. Under such approaches, parallel applications minimize the CPU utilization on the cloud whenever communication and synchronization between different parallel processes takes place. Also data center thus incur workloads in the cloud. A better approach should take responsiveness as the top priority so as to minimize the overhead at the data centers while assuring nontrivial effort from the side of the data centers. But at the same time it not only increases the communication cost but also results in the improper utilization of nodes in the data center with poor responses to parallel workload with many data center in the cloud. In this paper we propose an approach to improve the utilization of nodes and also respond to parallel workload in an efficient manner. We propose an approach called abstraction refinement scheduling for parallel job in multiple cloud centers. Under our approach, the problem of large-scale scheduling is solved using abstraction refinement schedulers. A challenging issue is how to solve the problem of scheduling for K-tier separation such that the scheduling is done fast with small abstraction refinements. On the basis of abstraction refinement schedulers, different schedulers are developed from various computing domains on simulated data centers. Extensive experiments are conducted to validate our approach where the results are implemented with CloudSim in JAVA on the parametric factors such as number of data centers, job assigned, time intervals, memory rate and CPU Cycles for abstraction scheduler.

Keywords: *Parallel Processing, Communication, Synchronization, Parallel Workload, Abstraction Refinement Scheduling*

1. INTRODUCTION

The cloud computing environment provides a cost-effective solution for the successful operation of business by applying virtualization technologies, distributed computing environment, as well as pay-what-you-use pricing mechanism. One of the trivial issues in cloud environment is the low utilization in data centers that are operated at 10 to 50 percent of their utilization level. Many research works has been contributed in this area for better utilization of data centers and therefore improving the productivity.

Priority-based Consolidation of Parallel Workloads (P-CPW) [1] used parallel scheduling algorithm aiming at improving the scheduling in a data center setting. The method was also proven to be efficient in providing consolidated parallel workload in data centers. Another method called, Multi-Objective Game Theoretic Scheduling (M-OGTS) [2] presented a communication and storage aware multi

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

objective algorithm to improve the cost and system level efficiency while scheduling.

A promising research direction which applies hyper heuristic scheduling [3] for cloud environments have attracted several research persons and contributed works on varied domains. Power allocation and load distribution is one of the most important issues across clouds and data centers. In [4], multi-variate optimization techniques were introduced to efficiently utilize the available resources with multiple servers. Global optimized scheduling algorithm using global greedy budget and gradual refinement [5] to minimize the cost during scheduling was studied in [6]. However, robustness to increased nodes remained unsolved.

The significance of robustness is obvious from the above discussion. In [7], multilayered graph modelling with the objective of ensuring robustness was introduced. However, to provide robustness on big data, resource scheduling and optimization framework was designed in [8] aiming at reducing the response time.

In this work, we present a novel approach called K-tier separation-based abstraction refinement scheduling for achieving high performance on multiple cloud centers for parallel job. We develop a Memory-based Abstraction Refinement Scheduling algorithm to measure the processor time and memory with abstraction refinement subject to constraint, resulting in abstract refinement with reduced memory consumption.

Our method differs from previous parallel workload methods found in the computer vision literature in that the proposed work apply K-Tier Separation for performing scheduling in a faster manner with small abstraction refinements. The K-Tier Separation specifies the scheduler condition and performs K-tier abstraction by applying a Best Fit Job Allocation which produces the CPU cycle for abstraction scheduler. By using the algorithmic developed here, we are able to attain a high performance on different time intervals over different computing domains on simulated data centers.

The paper is structured as follows. Section 2 reviews the relevant related work. In Section 3 the application and the K-Tier Separation-based Abstraction Refinement Schedulers, followed by the paper's problem definition is presented. Section 4 presents the experimental settings and parametric definitions. In Section 5, we validate and compare our algorithm against state-of-the-art woks through simulated and real-world experiments in a cloud environment. Section 6 concludes the paper.

2. RELATED WORKS

There have been many efforts made on the parallel job scheduling in cloud environment. In particular, multi-objective scheduling have been investigated by using virtualized clusters [9], assessing trustworthiness and competence [10], balanced partitioning algorithm [11]. Ordinal optimization method was applied to solve the multi-objective scheduling problem and selecting an ideal cloud service provider to assess the trustworthiness and competence. An approach based on ontology to analyze cloud service compatibility to simplify cloud service composition for unskilled users in cloud environment was investigated in [12].

There exists a huge body of literature for parallel computation [13] [14]. In [13], Exchanged Cross Cube was applied using optimal routing and broadcasting algorithms to reduce the communication overhead. To reduce the total energy consumption and scheduling delay during parallel computation, in [14], heterogeneity aware dynamic capacity provisioning scheme was presented.

Despite the fact that large number of schemes have been presented in the literature in recent years, a key challenge to address during parallel workload scheduling is the profit maximization with guaranteed quality of service.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In [15], a double resource renting scheme was designed to guarantee a service quality of all requests. Cloud computing framework is highly envisaged as an effective and efficient way of computing resources at a quicker rate. One attractive cloud computing framework is an optimal multi server configuration [16] which provided profit maximization and optimal server speed with optimal server size.

Energy efficient scheduling [17] and economic approach [18] for resource allocation in cloud environment was designed using heuristic algorithms and Back Propagation Neural Network (BPNN) ensuring parallel processing. However to address load distribution, in [19], special tasks with higher priority was addressed, reducing average response time. Energy-aware task scheduling [20] not only ensured optimized energy consumption but also maintained good performance in terms of make span.

In this paper, to overcome the shortcomings mentioned above, an abstraction refinement scheduler for parallel job in multiple cloud centers is presented which can guarantee the CPU cycle for abstraction scheduler and reduce the memory rate greatly. Moreover, a K-Tier Separation model is formulated for different schedulers from various computing domains which can prove to be more efficient than the optimal configuration in [1] [2].

3. K-Tier Separation-based Abstraction Refinement Schedulers

In this section, we describe our K-tier separation-based abstraction refinement scheduling approach for parallel job in multiple cloud centers formally through problem statement. We first discuss the abstraction refinement scheduling approach for parallel job and then the K-tier separation with small abstraction refinement.

3.1 Problem statement

Let us consider a graph '$G = (V, E)$' where 'V' represents the pieces of computation called jobs and 'E' represents the edges that transfers data between jobs. Each job has an assigned interval of time with each job an associated size. Let us further assume that there are 'K' resources in multi cloud data centers with their corresponding speed and memory given by '$S_i = S_1, S_2, \dots, S_n$' and '$M_i = M_1, M_2, \dots, M_n$'. The problem defined is as follows. The proposed approach designs data flow of parallel jobs with a data center holding a set of parallel jobs connected by communication links. The scheduling problem concerns assigning parallel jobs in multiple cloud centers and different schedulers developed from various computing domains on simulated data centers.

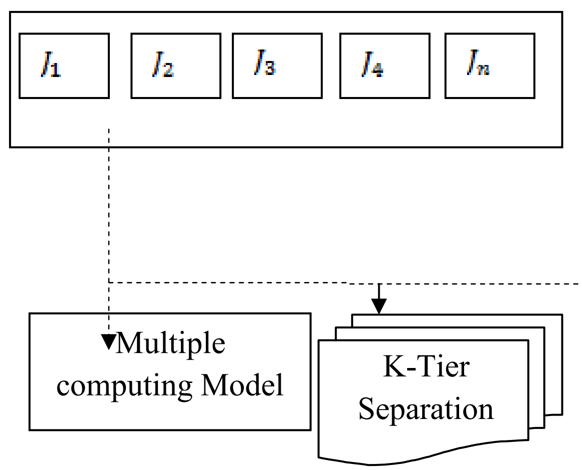

Figure 1 Block Diagram Of Abstraction Refine L Job In Multiple Cloud Centers

3.2 Abstraction refinement scheduling for parallel job

In this section, we propose an approach called abstraction refinement scheduling (ARS) for parallel job in multiple cloud centers. Under our approach, the problem of large-scale scheduling is solved using abstraction refinement schedulers .Figure 2 shows the block diagram of abstraction refinement scheduling for parallel job.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

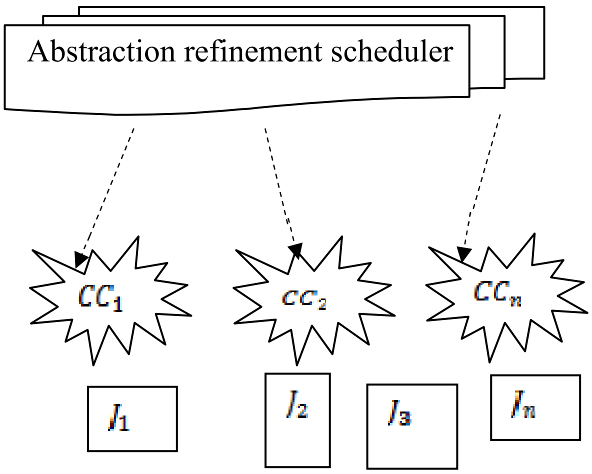

Figure 2 Block Diagram Of Abstraction Refinement Scheduling

Let us consider cloud centers '$CC_i = CC_1, CC_2, \ldots, CC_n$' to be assigned with the jobs '$J_i = J_1, J_2, \ldots, J_n$'. Then, as shown in the figure, using abstraction refinement scheduler, the process of scheduling of parallel jobs in cloud center is performed aiming at reducing the memory consumption during scheduling. An abstraction refinement scheduler initially solves the scheduling problem based on the abstract representations of parallel job sand resources required to accomplish the parallel jobs from multiple cloud centers. As the abstract representations are extensively smaller in size, scheduling of parallel jobs are done at an extremely faster rate, therefore reducing the memory consumption rate.

The abstraction refinement scheduler in abstraction refinement scheduling initially evaluates an abstract instance with the objective of concealing all those information that are highly irrelevantor redundant for obtaining a good schedule. In our proposed K-Tier Separation-based Abstraction Refinement Schedulers (KS-ARS), parallel jobs coming from multiple cloud centers with similar resource are grouped together. In this way, parallel jobs with similar resource are scheduled in bulk. Once the abstract instance is evaluated, the abstraction refinement scheduler updates the abstraction, resulting in a new abstract to be solved. This process is repeated until an optimal schedule is arrived at.

The abstraction refinement scheduling formalizes the notion of abstract instances of scheduling for parallel job problems where an abstract instance represents a refinement of another one. The abstraction refinement scheduler maintains fixed abstraction of data center and schedules parallel job to group of nodes '$N_i = N_1, N_2, \ldots, N_n$' waiting to be assigned with. If job 'J_i' is assigned to run in a resource of speed 'S_i' and memory 'M_i', with each job associated with a time factor 'T_i', then the processor time and the memory required to complete a job 'J_k' is mathematically formulated as given below

$$Processor\ Time = \frac{T_k}{S_k} \tag{1}$$

$$Memory = \frac{M_k * T_k}{S_k} \tag{2}$$

In a similar manner, the actual memory to finish all the jobs in a parallel manner in multi cloud centers is as given below.

$$Memory_n = \frac{\sum_{k=1}^{n} M_k * T_k}{S_k} \tag{3}$$

From (3), 'n' is the set of all jobs assigned to resource 'k' in multi cloud centers. From (3), the main objective of abstraction refinement scheduling is to minimize the product of memory and time incurred for all the parallel jobs in multi cloud centers and is mathematically formulated as given below.

$$Min\ (P) = Memory_i * Time_i$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Where 'i belongs to the resources available in multi cloud centers for scheduling. With this, the problem of large-scale scheduling (i.e. parallel jobs) is solved using abstraction refinement schedulers reducing the memory consumption.

The actual advantage of abstraction refinement scheduler principle is the capability to handle in various computing domains. The proposed KS-ARS uses a K-tier separation that performs scheduling with small abstraction refinements. Figure 3 shows the algorithm form of Memory-based Abstraction Refinement Scheduling.

Input: Speed '$S_i = S_1, S_2, \ldots, S_n$', Memory '$M_i = M_1, M_2, \ldots, M_n$', Cloud Centers '$CC_i = CC_1, CC_2, \ldots, CC_n$', Jobs '$J_i = J_1, J_2, \ldots, J_n$',
Output: Optimized memory consumption
Step 1: Begin Step 2: For each job 'J_i' Step 3: For each cloud center 'CC_i' Step 4: Measure processor time to complete a job using (1) Step 5: Measure memory to complete a job using (2) Step 6: Measure memory to finish jobs in parallel using (3) Step 7: Perform abstraction refinement scheduling subject to constraint using (4) Step 8: End for Step 9: End for Step 10: End

Figure 3 Memory-Based Abstraction Refinement Scheduling Algorithm

As shown in the figure, the Memory-based Abstraction Refinement Scheduling algorithm first measures the processor time to complete single job in each cloud center. Next, the memory required to accomplish the job is measured. Based on the memory, the abstraction refinement scheduling is performed, aiming at optimizing the memory consumption during scheduling. Finally, memory required for abstraction refinement scheduling for parallel job in multiple cloud centers are measured.

3.3 K-Tier Separation

In this section, the challenging issue to solve the problem of scheduling for K-tier separation with small abstraction refinements is presented. On the basis of abstraction refinement schedulers, different schedulers are developed from various computing domains on simulated data centers.

K-Tier Separation (i.e. K-tier data center scheduler) starts with initial abstract job obtained from input job by using abstraction refinement scheduler. Initial abstract data center is obtained from input data structure by combining all the nodes into a single node. The K-tier data center scheduler keeps job abstraction constant whereas on the other hand refines data center abstraction whenever required.

The K-Tier Separation algorithm takes a job 'J_i', a cloud center 'CC_i' and condition on scheduler '$Min\ (P)$' as input. The scheduler condition '$Min\ (P)$' check whether a schedule meets a given deadline. Memory allocator and processor time maintains a partition of the memory and processing time interval with the objective of identifying the best suitable free memory block (satisfying the objective constraint obtained from (4). In a similar manner, the allocation of memory and processor time for K-Tier subject to the constraint is as given below.

$$KTA = Min\ (J_i) \sum_{i=1}^{K} Memory_i * Time_i \quad (5)$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Where 'KTA' symbolizes the K-Tier Abstraction and 'K' symbolizes the schedulers. From (), the job waiting in multiple cloud centers with the memory and time as constraint is taken as the factor for performing K-Tier Abstraction. Each refinement step further partitions the memory block into two other new memory and processing time blocks and is formulated as given below.

$$\{(Time_i, \ldots, Time_{i+j}), \ldots, (Time_{i+j+1}, \ldots, Time_K)\} \quad (6)$$

When an allocated memory block is freed, compaction is performed so that new nodes enter into the cloud center with the objective of assigning and scheduling its job in a parallel fashion. The parallel allocation of jobs in the proposed KS-ARS is performed using Best Fit Parallel Job Allocation model that efficiently schedules the jobs from one job to nodes close to each other.

$$BF = \sum_{i=1}^{K} Min\ Vicinity(N_i) \quad (7)$$

On the other hand, the data center representation changes with each allocation of jobs performed in a parallel fashion. Figure shows the algorithmic flow for Abstraction Scheduler.

Input: Job '$J_i = J_1, J_2, \ldots, J_n$', cloud center '$CC_i = CC_1, CC_2, \ldots, CC_n$', scheduler condition '$Min\ (P)$', Speed '$S_i = S_1, S_2, \ldots, S_n$', Memory '$M_i = M_1, M_2, \ldots, M_n$'
Output: Optimized CPU Cycles for abstraction scheduler
Step 1: Begin Step 2: For each job 'J_i' Step 3: For each cloud center 'CC_i' and different schedulers Step 4: Perform K-Tier Abstraction using (5) Step 5: Perform Abstraction refinement using (6) Step 6: Evaluate Best Fit job allocation using (7) Step 7: End for Step 8: End for Step 9: End

Figure 4 Abstraction Scheduler algorithm

As shown in the figure, the scheduler algorithm based on CPU cycle, initially performs K-Tier Abstraction based on the memory and processor time. Next, the abstraction refinement is performed according to the job to be served or not, the processes do not have to wait because a job which occupies the minimum of memory and processor time (CPU) is assign first, then the node closest to it and so on. On the other hand, the CPU is not idle while there are parallel jobs to be executed. This in turn optimizes the CPU Cycles for abstraction scheduler.

4. EXPERIMENTAL SETTINGS

K-Tier Separation-based Abstraction Refinement Schedulers (KS-ARS) uses the Cloud Sim simulator to work under the simulation environment. The experimental work is carried out in JAVA language for evaluating the parallel job scheduling in multiple cloud centers. The Cloud Sim simulator toolkit has been used as a simulation platform with 8 GB of RAM and 1 TB of storage space. Amazon Access Samples dataset information is used on the transaction processing between cloud users and cloud servers.

The information included in the Amazon Access Samples dataset comprises of dense dataset where less than 5% of the attributes were used for evaluating the parallel job scheduling in multiple cloud centers using the proposed K-Tier Separation-based Abstraction Refinement Schedulers. The Amazon Access Samples dataset includes four categories of attributes including Person_Attribute, Resource_ID, Group_ID and System_Support_ID.

Experiments are conducted on several cloud users to identify the performance level of parallel job scheduling against the existing methods. The K-Tier Separation-based Abstraction Refinement Schedulers (KS-ARS) is

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

compared against the Priority-based Consolidation of Parallel Workloads (P-CPW) [1] and Multi-Objective Game Theoretic Scheduling (M-OGTS) [2] in cloud environment. Experiment is conducted on factors such as number of data centers, job assigned, time intervals, memory rate and CPU Cycles for abstraction scheduler.

The scheduler time intervals measures the time required for scheduling each jobs and the time required to schedule each job. Therefore the scheduler time interval is mathematically evaluated as given below.

$$ST = \sum_{i=1}^{n} J_i * Time\ for\ each\ job \quad (8)$$

Where 'ST' symbolizes the scheduler time interval. The scheduler time interval is measured in terms of milliseconds (ms). The memory required for scheduling each job and the total number of jobs assigned in multi cloud center is the actual memory rate. The memory rate is measured in terms of kilo bytes (KB) and is formulated as given below.

$$M = \sum_{i=1}^{K} J_i * Memory_i \quad (9)$$

Where 'M' refers to the memory rate measured using the 'J_i' job assigned, with respect to memory consumption '$Memory_i$' for scheduling each job, in multi cloud center. Lower the memory rate more efficient the method is said to be. CPU Cycles for abstraction scheduler or node utilization is the ratio of jobs assigned to the cloud center. It is mathematically formulated as given below.

$$NU = \sum_{i=1}^{n} \frac{J_i}{CC_i} \quad (10)$$

Where 'NU' symbolizes the node utilization, obtained using 'J_i' jobs assigned with respect to the cloud centers, 'CC_i' respectively.

5. Discussion

The result analysis of K-Tier Separation-based Abstraction Refinement Schedulers (KS-ARS) is compared with existing K-Tier Separation-based Abstraction Refinement Schedulers (KS-ARS) is compared against the Priority-based Consolidation of Parallel Workloads (P-CPW) [1] and Multi-Objective Game Theoretic Scheduling (M-OGTS) [2] in cloud environment respectively. Table 1 represents the scheduler time interval with different number of jobs assigned using CloudSim simulator and comparison is made with two other methods, namely P-CPW [1] and M-OGTS [2].

Table 1 Tabulation For Scheduler Time Interval

Job assigned	Scheduler time interval (ms)		
	KS-ARS	P-CPW	M-OGTS
5	5.38	5.52	5.61
10	7.85	10.65	13.35
15	9.14	12.04	15.04
20	12.32	15.12	18.02
25	15.89	18.69	21.39
30	19.32	22.12	25.02
35	24.16	27.06	30.01

To ascertain the performance of the Scheduler time interval, comparison is made with two other existing methods Priority-based Consolidation of Parallel Workloads (P-CPW) [1] and Multi-Objective Game Theoretic Scheduling (M-OGTS) [2] respectively. In figure 5, the parallel jobs assigned in multi cloud centers are varied between 5 and 35 at different time intervals using Amazon Access Samples dataset.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

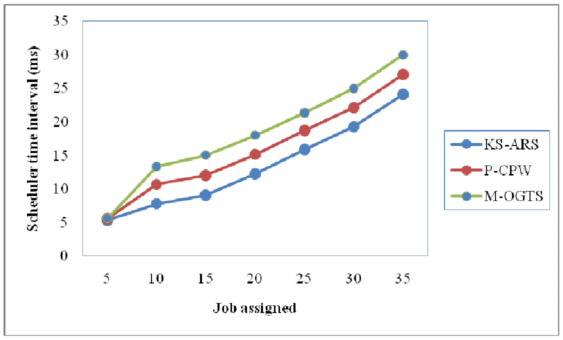

Figure 5 Measure Of Scheduler Time Interval

From the figure it is illustrative that the scheduler time interval is lower or decreased using the proposed KS-ARS when compared to the two other existing methods. This is because with the application of the abstraction refinement scheduling (ARS) for parallel job in multiple cloud centersthat uses abstraction refinement schedulers for efficient scheduling of parallel jobs. Based on the abstract representations of parallel jobsand resources required to accomplish the parallel jobs, in turn reduces the scheduler time interval by 19.54% compared to P-CPW.

Furthermore, the evaluation of for concealing all those information that are highly irrelevantor redundant for obtaining a good schedule reduces the scheduler time interval by 39.06% compared to M-OGTS respectively. The comparison of memory consumption during scheduling s presented in table 2 with respect to the job assigned ranging from 5 to 35 taken up for experimental purpose. With difference noted in the job assigned with each job differing in kilo bits sent, we find a gradual increase in scheduling time interval, but KS-ARS has proved to be better than the two state-of-the-art works.

Table 2 Tabulation For Memory Rate

Job assigned	Memory rate (KB)		
	KS-ARS	P-CPW	M-OGTS
5	61	67	72
10	85	91	100
15	104	110	107
20	115	123	132
25	128	135	145
30	142	152	159
35	155	168	175

In figure 6, we depict the memory rate with parallel jobs assigned in the range of 5 to 35 for experimental purposes. From the figure, the memory rate during scheduling in multi cloud centers using the proposed KS-ARS is lower when compared to two other existing methods P-CPW [1] and M-OGTS [2] respectively.

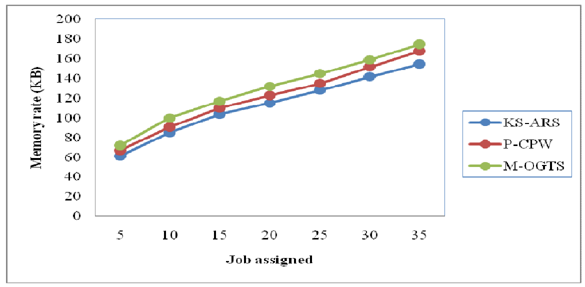

Figure 6 Measure Of Memory Rate

From figure 6 it is illustrative that the memory rate for scheduling using KS-ARS is reduced because the method uses an Abstraction Scheduler algorithm where the parallel jobs are allocated in multi cloud centers using Best Fit job allocation. In Abstraction Scheduler algorithm, parallel workloads are assigned using K-Tier Abstraction based on the memory and processor time. Therefore, the memory for scheduling with respect to job assigned in multi cloud centers is reduced by 7.21% compared to P-CPW. With this K-Tier Abstraction, when an allocated memory block is freed linear, compaction is performed with the aim of reducing the memory rate by 14.44% compared to M-OGTS. The node utilization of KS-ARS is elaborated in table 3. KS-ART was considered with 70 different cloud centers for experimental purpose using CloudSim simulator.

Table 3 Tabulation For Node Utilization

Number of cloud centers	Node utilization		
	KS-ARS	P-CPW	M-OGTS
10	81.35	69.85	59.72
20	83.85	71.21	61.21
30	85.17	73.32	73.32

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

40	75.29	63.43	63.43
50	79.17	67.32	67.32
60	80.23	68.32	68.32
70	84.89	72.12	72.12

Figure 7 shows the measure of node utilization with respect to different cloud centers in the range of 10 to 35. With the increase in the number of cloud centers, the node utilization is also increased, though not in a linear manner due to the different cloud centers performing different job size, though efficiency proved to be higher in a comparative manner using the KS-ARS method.

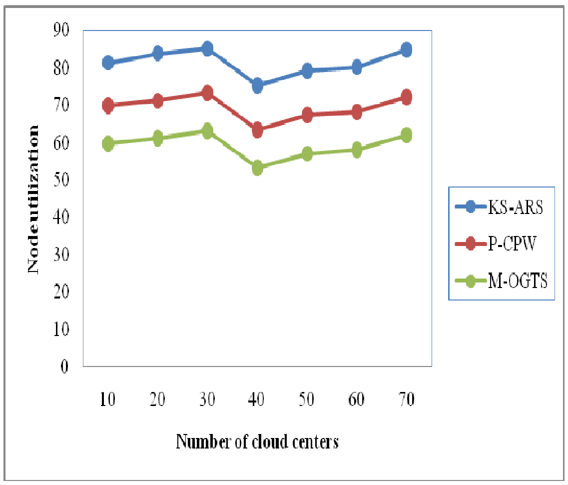

Figure 7 Measure Of Node Utilization

As illustrative in the figure, the node utilization is observed to be high using KS-ARS. This is because of the application of Memory-based Abstraction Refinement Scheduling algorithm, where the allocation of memory and processor time for K-Tier keeps job abstraction constant with the arrival of parallel workload at different time. Instead of scheduling the parallel job in multiple cloud centers in a random manner, CPU Cycles for abstraction scheduler or node utilization is made in an efficient manner resulting in the improvement of node utilization in KS-ARS method by 14.81% compared to P-CPW. In addition, the KS-ARS method uses evaluates an abstract instance model and as a result, the node utilization is improved by 27.33% compared to M-OGTS.

6. CONCLUSION

In this paper we propose a K-tier separation-based abstraction refinement scheduling approach for parallel job in multiple cloud centers. The scheduling of parallel job in multiple cloud centers is formulated as an optimal node utilization problem where the scheduling of parallel job is performed based on the abstract instance. By measuring the abstract instance, we propose a Memory-based Abstraction Refinement Scheduling algorithm based on the parallel jobs coming from multiple cloud centers with similar resource are grouped together. Next, the problem of scheduling K-tier separation is performed on the basis of abstraction refinement schedulers. Extensive simulation is carried out to evaluate the proposed parallel job scheduling in multi cloud center. The results validate the effectiveness of the proposed method and show that it significantly outperforms two traditional parallel workload scheduling in cloud environment.

REFERENCES

[1] Xiaocheng Liu, Chen Wang, Bing Bing Zhou, Junliang Chen, Ting Yang, and Albert Y. Zomaya, "Priority-Based Consolidation of Parallel Workloads in the Cloud", IEEE Transactions on Parallel And Distributed Systems, Volume 24, Issue 9, September 2013, Pages 1874-1883.

[2] Rubing Duan, Radu Prodan, and Xiaorong Li, "Multi-Objective Game Theoretic Scheduling of Bag-of-Tasks Workflows on Hybrid Clouds", IEEE Transactions on Cloud Computing, Volume 2, Issue 1, January-March 2014, Pages 29-42.

[3] Chun-Wei Tsai, Wei-Cheng Huang, Meng-Hsiu Chiang, Ming-Chao Chiang, and Chu-Sing Yang, "A Hyper-Heuristic Scheduling Algorithm for Cloud", IEEE Transactions on Cloud Computing, Volume 2, Issue 2, April-June 2014, Pages 236-250.

[4] Junwei Cao, Keqin Li, and Ivan Stojmenovic, "Optimal Power Allocation and Load Distribution for Multiple Heterogeneous

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Multicore Server Processors across Clouds and Data Centers”, IEEE Transactions on Computers, Volume 63, Issue 1, January 2014, Pages 45-58.

[5] Yang Wang and Wei Shi, “Budget-Driven Scheduling Algorithms for Batches of Map Reduce Jobs in Heterogeneous Clouds”, IEEE Transactions on Cloud Computing, Vol. 2, No. 3, July-September 2014, Pages 306-319.

[6] Marco Polverini, Antonio Cianfrani, Shaolei Ren, and Athanasios V. Vasilakos, “Thermal-Aware Scheduling of Batch Jobs in Geographically Distributed Data Centers”, IEEE Transactions on Cloud Computing, Volume 2, Issue 1, January-March 2014, Pages 71-84.

[7] Kashif Bilal, Marc Manzano, Samee U. Khan, Eusebi Calle, Keqin Li, and Albert Y. Zomaya, “On the Characterization of the Structural Robustness of Data Center Networks”, IEEE Transactions on Cloud Computing, Volume 1, Issue 1, January-June 2013, Pages 64-77.

[8] Dawei Sun , Guangyan Zhang , Songlin Yang , Weimin Zheng , Samee U. Khan, Keqin Li, “Re-Stream: Real-time and energy-efficient resource scheduling in big data stream computing environments”, Elsevier, Information Sciences, Volume 319, 20 October 2015, Pages 92–112.

[9] Fan Zhang, Junwei Caob, Keqin Li, Samee U. Khand, Kai Hwang, “Multi-objective scheduling of many tasks in cloud platforms”, Elsevier, Future Generation Computer Systems, Volume 37, July 2014, Pages 309–320.

[10] Nirnay Ghosh, Soumya K. Ghosh, and Sajal K. Das, “SelCSP: A Framework to Facilitate Selection of Cloud Service Providers”, IEEE Transactions on Cloud Computing, Volume 3, Issue 1, January-March 2015, Pages 66-79.

[11] Xiaochun Yun, Guangjun Wu, Guangyan Zhang, Keqin Li, and Shupeng Wang, “FastRAQ: A Fast Approach to Range-Aggregate Queries in Big Data Environments”, IEEE Transactions on Cloud Computing, Volume 3, Issue 3, April-June 1 2015, Pages 206 – 218.

[12] Amir Vahid Dastjerdi, and Rajkumar Buyya, “Compatibility-Aware Cloud Service Composition under Fuzzy Preferences of Users”, IEEE Transactions on Cloud Computing, Volume 2, Issue 1, January-March 2014, Pages 1-13.

[13] Keqiu Li, Yuanping Mu, Keqin Li, and Geyong Min, “Exchanged Crossed Cube: A Novel Interconnection Network for Parallel Computation”, IEEE Transactions on Parallel and Distributed Systems, Volume 24, Issue 11, November 2013, Pages 2211-2219.

[14] Qi Zhang, Mohamed Faten Zhani, Raouf Boutaba, and Joseph L. Hellerstein, “Dynamic Heterogeneity-Aware Resource Provisioning in the Cloud”, IEEE Transactions on Cloud Computing, Volume 2, Issue 1, January-March 2014, Pages 14-28.

[15] Jing Mei, Kenli Li, Aijia Ouyang and Keqin Li, “A Profit Maximization Scheme with Guaranteed Quality of Service in Cloud Computing”, IEEE Transactions on Computers, Volume 64, Issue 11, November 2015, Pages 3064 – 3078.

[16]Junwei Cao, Kai Hwang, Keqin Li and Albert Y. Zomaya, “Optimal Multi server Configuration for Profit Maximization in Cloud Computing”, IEEE Transactions on Parallel and Distributed Systems, Volume 24, Issue 6, June 2013, Pages 1087 – 1096.