Indonesian Journal of Electrical Engineering and Computer Science
Vol. 11, No. 3, September 2018, pp. 1102~1108
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v11.i3.pp1102-1108

Review of Advancements in Multi-tenant Framework in Cloud Computing

Suresh K[1], Jagadeesh Kannan R[2]
[1]School of Computer Science and Engineering, VIT University, Vellore, India
[2]School of Computer Science and Engineering, VIT University, Chennai, India

Article Info

Article history:

Received May 9, 2018
Revised Jun 2, 2018
Accepted Jun 21, 2018

Keywords:

Cloud computing
Multi-tenancy
Cloud data

ABSTRACT

As the cloud computing is gaining more user base the problem of simultaneously catering computational resources to multitude of users or their application is on rise. It remains a critical problem and pose hindrance in scalability of cloud computing. Thus, in order to layout the proper solution for the mentioned problem; it is necessary to sum up a proper knowledge based of the existing solution, there drawbacks and a detail analysis of its performances. In this study we present a review of multi-tenant frameworks and approaches used in the industry which reaps advantages to facilitate multi-tenancy.

Corresponding Author:

Suresh Kumar,
School Of Computer Science And Engineering,
Vellore Institute Of Technology (VIT) University,
Vellore, Tamilnadu, India.
Email: dsureshkumar4u@gmail.com

1. INTRODUCTION

Cloud computing remains a standout technological solution amongst the most innovative encouraging modern advancements, on which the scientific community has embarked [1-3]. One of its best feature i.e., Multi-tenancy is one of the key highlights of cloud computing. In regular framework of cloud computing, the service suppliers offer a committed cloud resource to the inhabitant application/programs (clients), where no information is intermixed with different inhabitants. This type of model does not add adaptability to cloud administrations and financial aspects of its scalability. Then again, in multi-tenant cloud computing, frameworks, applications, and database are meant to be shared among every one of its inhabitants. But this has a drawback that its inhabitants will most likely be unable to modify their utilization of cloud resources with a specific end goal to fit their particular needs. Multi-tenancy can be partitioned into two sorts:

(i) For Multi-tenancy based on Infrastructure-as-a-Service (IaaS), its occupants has the ability to provision computing resources, storing for latter use or any other benefits with network resources. Accordingly, a service provider must permit its inhabitants for virtualization and asset sharing to achieve multi-tenancy. The advantages caters by this type of multi-tenancy is expanded use of hardware and software resources and simplicity of support, versatility/flexibility, i.e., management of dynamic workload comprising of huge quantities of inhabitants and is taken care of by scaling up/down of computing resources.
(ii) In Software-as-a-Service (SaaS), a solitary case of facilitated applications is utilized by various end users or clients at the same time, e.g., Force.com [4].

Despite the fact of multi-tenancy permits cloud providers to better use computing assets/resources, supporting the improvement of more adaptable administrations in view of economy and energy savings, by

lessening infrastructural costs, though how to successfully achieve this is a principal question. For example, the implementation of multi-tenant cloud computing pose following difficulties:

(a) Versatility,
(b) Asset re-provisioning (e.g., taking care of the demand of huge volumes of occupants per asset)
(c) Customization (per-inhabitant benefit customization) [5-7].
(d) Essentially it should be dynamic in nature, or polymorphic, to satisfy the individual desires of different inhabitants and their clients [4].

The best possible and productive planning can have critical effect on the execution of the work flow. All in all, scheduling for such a task was shown to be a NP-difficult issue [8]. Along these lines, there is no ideal arrangement inside polynomial time. Heuristic algorithms were broadly created keeping in mind the end goal to achieve an optimal configuration [9-12]. In any case, there is no specific asset administration structure for scheduling computational work flow in multi-tenant cloud computing situations [13-15].

This study presents a survey of computational advances in multi-tenant cloud computing condition, outlining all sorts of ways to achieve multi-tenancy for resource management processes in order to accommodate IaaS or SaaS. The rest of the paper is organized as takes after. In section 2 we present the architecture and methods of multi-tenancy in more prominent detail. Section 3 portrays the remarkable highlights of previous implementations, and lastly, we finish up with conclusion in Section 4.

2. LITERATURE REVIEW

2.1 Architecture of Multi-tenant Framework

There are a plenty of systems for workflow administration and one of the prime example is Pegasus [16]. In any case, a significant number of their highlights are streamlined for conventional grid based framework and cluster based computing in order to execute serial or parallel computational jobs and in this manner will be unable to achieve essential features of distributed computing, wherein such frameworks are tend to experience the ill effects of constrictive mode of asset or resource provisioning. In spite of the fact that there are few works tending to cater work flow scheduling on demand [17]. Given the rise of assorted arrangements of logical flow of work process of multi-tenant applications are given to deal with a variety of condition owing to its dynamic nature. Thus, a multi-tenant framework should be adaptive with respect to varying stages of computational work flows which are expected to lessen the operational cost.

This sort of design empowers such flow of work process in multi-tenant applications to share a platform while exploiting the resources in versatile manner and is dependent on pay-as-you-go based charging model. Its architecture composed of 4 types of layers. The primary layer comprises of job creator. The second layer, which is a middleware component, comprises of job dispatcher, manage service queue, share and allocate resources. The third layer is a form of a virtual infrastructure layer, while the fourth layer comprises of the physical framework layer. Multi-tenant architectures can be arranged in various ways. In this segment discuss about five of the most recognizing properties: the application class, power/performance, processing elements, memory system, and accelerating agents/integrated peripherals. The elements of multi-tenant framework are described below.

Enormous intricate related issues have emerged, which have cropped-up that must be addressed to use clouds the purpose tat which they must be used in designing and implementation. However, comparing to other parameters fault tolerance and data securing which stored on the cloud are considered important [53]. The designed main memory structure in common have used the indexed of different tenants in a single tree in achieving the effects on memory space complexity. Meanwhile, in the cloud tenant placement algorithm is been extracted. The similar tenants are placed with the similar indexes in the same particular nodes for further optimization of the memory space [54]. The FADE based upon the cryptographic key arrangements which are in self kept up by a superior account of key supervisors which has free of outsider mists. In concern, FADE comes on an overlay frameworks supported that the works flawlessly gets on today's distruibuted storage administrations [55]. A programmatic access control mechanism is supporting particular sharing of compositive electronic health records (EHRs). The approach guarantees privacy issues which are accommodate for the processing accessible results for the clients [56].

2.1.1 Application Class

In the event that a cloud computing e is focused to a specific multi-tenant framework space, the construction modeling can be made to mirror this. The outcome is an outline that is productive for the space being referred to however regularly ill-suited to different territories in cloud computing. The compelling case is an ASIC. Tuning to a multi-tenant framework area can have a few positive results. Maybe the most profitable is the potential for critical power reserve funds [18]. Traditional DSPs are a decent illustration.

There are two wide classes of processing into which a multi-tenant framework can fail: information processing ruled and control overwhelmed.

2.1.2 Information Processing Dominated

Information processing-commanded based multi-tenant framework contains numerous natural sorts of applications including graphics rasterization, image processing, sound processing, and remote baseband processing. A considerable lot of signals processing calculations are a piece of this cluster [19]. The calculation of these sorts of applications is commonly a grouping of operations on a stream of information with practically no information reuse. The operations can every now and again be performed in parallel and regularly require high throughput and performance to handle a lot of information. These sorts of applications support outlines that have the same number of processing elements as handy with respect to coveted power/performance proportion.

2.1.3 Control Processing Dominated

Control-overwhelmed multi-tenant framework incorporates document compression/decompression, system processing, and value-based job processing. The code for these sorts of framework has a tendency to be ruled by restrictive branches, confounding parallelism [20]. The projects themselves frequently need to monitor a lot of state and regularly have a high measure of information reuse. These sorts of applications support a more unassuming number of universally useful processing elements to handle the unstructured way of control commanded code.

In all cases, no multi-tenant framework can fit into these divisions, yet execution periods of a multi-tenant framework might give reasonable performance. For example the H.264/AVC video codec is information predominant while performing the block filter, yet control commanded while compacting or decompressing video utilizing context-adaptive binary arithmetic coding (CABAC) pressure. It is profitable to consider applications falling into these divisions to see how distinctive multi-tenant frameworks outline angles can influence performance [21]. An uneven building design might do exceptionally well on the information overwhelmed part of the H.264/AVC, however be extremely wasteful for CABAC encoding/decoding, prompting less than the exactly craved performance.

2.1.4 Processing Elements

In this area, we cover the architecture and multi-tenant frameworks and its processing component. The architecture, or all the more instruction set architecture (ISA), characterizes the equipment software interface.

2.1.5 Architecture

In traditional multi-tenant processors, the ISA of every core is commonly a legacy ISA from the comparing uni-processor with minor adjustments to bolster parallelism, for example, the expansion of nuclear instructions for synchronization. The points of interest to legacy ISAs are the presence of usage and the accessibility of programming devices [22]. An ISA might likewise be specially defined.

ISAs can be named reduced instruction set computer (RISC) or complex instruction set computer (CISC). In spite of the fact that this was a disputable qualification times requisite in today's needs, the multi-tenant frameworks refinements have been obscured: most CISC machines look all that much like their RISC partners once decoding has been finished. On the code front, the distinctions are still particular. CISC has the edge in code size because of the more noteworthy determination of instructions and wealthier semantics accessible. RISC, then again, has bigger code sizes because of the need to copy more complex instructions with the smaller set of RISC instructions. The benefit of RISC is it gives a less demanding focus to compilers and considers less demanding micro architectural outline [23].

Past the base application of the ISA, cloud service providers have been ceaselessly adding ISA expansions to enhance performance for regular operations. Intel has included MMX, MMX2, and SSE1-4 to enhance sight and sound performance. ARM has included comparative instructions for media with its NEON instruction set. These instructions consider a superior performance/power utilization proportion as specific equipment can do operations like a vector transpose in one instruction. The delicate core supplier Tensilica has made this the primary offering point for their Xtensa CPUs, offering adjustable uncommon purpose instructions for specific plans [24].

2.1.6 Architecture Of Multi-Tenant Frameworks For Micro Services

The processing component micro services administer, in numerous regards, the performance and power utilization that can be normal from the multi-tenant. The micro services of every processing component are regularly customized to the multi-tenant framework which is focused by the parallel finite

state machine. In spite of the fact that the business offerings of the significant chip makers like Intel utilize quantities of indistinguishable cores into a homogeneous architecture, it is regularly favorable to join distinctive sorts of processing elements into a heterogeneous architecture. The focus is again to acquire a power advantage without loss of performance [25]. A commonplace association has a control multi-tenant framework marshaling the exercises of an outfit of more straightforward "information plane" cores. In information ruled applications, such architectures can frequently give elite at low power. The downside is that the programming model for heterogeneous architectures is a great deal more entangled.

The least complex sort of processing component is the all together processing component. This kind of processing component decodes and executes instructions in system request and powerfully represents information sending and control dangers. There are two principle performance parameters that can be altered to get the fancied performance. Initially, numerous pipelines can be added to bring and issue more than one instruction in parallel, making a superscalar processing component to expand performance [26]. Be that as it may, expanding issue width requires additional rationale to give more complex information sending ways and peril recognition to guarantee right code execution in the pipelines.

The complexity of the rationale becomes more prominent than quadratic ally with the quantity of pipelines, and a state of unavoidable losses. It attempts different things with universally useful applications recommend that indicate what is around three four pipelines, obviously this is exceptionally subject to the applications. Secondly, performance can likewise be enhanced by expanding the quantity of pipeline stages in multi-tenant frameworks, in this way lessening the processing involved in each stage. This empowers a speedier clock to the detriment of more noteworthy penalties if the instruction succession is broken by branches. All together elements have little opportunity to exit the bucket region, low power, and are effectively consolidated in expansive numbers if amulti-tenant framework has plentiful thread level parallelism (TLP) and few performance delicate serial segments. For instance, NVIDIA's G200 group's together 240 altogether cores in light of the fact that graphics processing is exceedingly parallel with couple of serial segments [27].

Taking the superscalar core further to pick up however much single thread performance as could reasonably be expected is the out-of-bounds of the architecture. It endeavors to powerfully find and plan numerous instructions "out of request" to keep the pipelines full. The dynamic form of scheduling requires exceptionally complex and power hungry hardware to monitor all in-flight instructions. Out of bound request composed cores are most suitable for applications that have an extensive variety of practices and elite is required. Then again, the logical complexity implies this sort of processing component is not power effective and requires significant bite the dust range. Most of the out-of-request processes are multi-issue, as single-issue out-of-request processors don't have much point of interest over a less complex all together core. Since the out-of-request core is huge and power hungry, not very many can be joined practically speaking. On the other hand, they are ideal if the applications to be run are control commanded and have huge basic serial partitions and direct TLP [28]. For instance, the ARM Cortex A9 is focused for netbook computers, and requires single thread performance over TLP, so it uses a modest bunch of out of request cores.

To build performance over superscalar architectures, yet wipe out the complexity of the additional logical expression is expected to legitimately execute the instruction stream, single-instruction, multiple data (SIMD) or long instruction word (VLIW) architectures can be utilized. The SIMD architecture makes utilization of wide registers split into paths to prepare multiple data focuses with one instruction. A basic case is the expansion of two vectors component savvy. Every pair of elements is handled in its own particular path. This style of architecture is appropriate for data concentrated applications that are data parallel. A case is the IBM Cell [23] that uses numerous SIMD cores focused on towards data overwhelmed applications. SIMD architecture is very wasteful for broadly useful processing.

To abstain from being restricted to one instruction processing multiple data focuses, a VLIW can be utilized. VLIW utilizes multiple pipelines yet does not normally have the sending, planning, and peril discovery rationale of a superscalar core. Rather, the compiler is depended upon to gathering instructions into bundles that can be executed in parallel and ensure no data or control dangers—the complexity has been moved to the compiler. VLIW execution takes into consideration wide machines that can handle multiple data focuses with multiple instructions in the meantime, giving it a particular point of interest over SIMD. In any case, VLIW can endure extreme underutilization issues if the compiler can't discover adequate parallelism. VLIW and SIMD are both elite and power-productive outlines yet are normally appropriate for just certain sorts of multi-tenant framework codes with substantial quantities of independent operations that can found by compilers or the programmer.

2.2 Scheduling In Multi-Tenant Framework

The issue of scheduling the computational workload had been extensively researched to achieve effective parallel and distributed computing. In past years the cloud computing has also catch up with the

ongoing research to come up with the ways of effective workload scheduling and resource allocation. The method used in this context are widely dependent on the nature of workload, frequency of job allocation and availability of resources both hardware & software [29]. The following section depicts the most notable used for infrastructure management in cloud computing.

2.2.1 Heuristic Algorithm

Some of the most recognizable example of heretic method are scheduling of list, duplication of task and clustering. Among the list scheduling approach the methods like include Heterogeneous Earliest-Finish-Time (HEFT) [30] and Fast Critical Path (FCP) [31] are mainly used to deploy single workflow of computational load. It is most effective when used in combination with multi-objective optimization method [32]. Also, in the context of task duplication based approach the method known as task duplication for heterogeneous system is the most prominent one [33, 34]. Lastly for clustering or grouping based heuristics CASS-II is mainly used for task clustering when deployed over heterogeneous system [35, 36].

In order to minimize the execution time a cost and time based heuristic was also developed [37]. For parallel execution of task the scheduling technique based on Whittle's index was used [38]. Since, the main emphasis is on minimizing the cost of time and migration of resource thus there raise the requirement to develop an opportunity based scheduling method. Owing to its dynamic nature a dynamic resource allocation technique was introduced in [39]. Here, the allocation of resources for dynamic scenario is vastly improved. A variant of this method known as best fit scheduling was presented in [40]. In another study, it was presented to use a static asset allocation over a multi-tenant scheduling which had considering process and time delay [41]. Nonetheless, this technique may not be appropriate when put in regards for multi-tenancy and cloud computing conditions owing to its delay, and erratic execution [42].

2.2.2 Meta-Heuristic Algorithm

This method included the utilization of warm based approach which is also known as particle of swarm optimization or PSO [43, 44]. This limits the cost of execution in distributed setting. Other variants of this approach includes evolutionary algorithms, ant colony optimization etc [45]. In another study the problem of dynamic scheduling was overcome by the utilization of a pricing model and was compared with pare to-optimal method. This approach was based on multi-objective evolution and among the most notable includes SPEA2 & NSGA-II. It is to be noted that heuristic based approach is suitable for only certain specific problems for example in situation involving simple organization of the job workflow; whereas the meta-heuristic based method is quite adaptable and its applicability varies over a wide range of problems. Though they are very time consuming methods for scheduling job workflow involving huge number of jobs but it often used to derive an optimal solution for grid computing.

2.2.3 Scientific Workflows Execution

In the study mentioned in [46] the author had studied and compare the performance and cost of operation for several jobs involving with scientific workflow. It is concluded form the study that due to the absence of parallel database system the resource provided by Amazon cloud computing environment also known as EC2 isn't suitable for coping with job operation involving intensive I/O operations. Latter, in order to solve the problem a data locality based task scheduling algorithm was developed by the authors of [47]. Even though the algorithm provides sufficient performance but the scalability issue remains ineffective. In a similar fashion matrix and k-means based strategies were adopted for determining placement of data [48].

2.2.4 Deadline-Aware Scheduling

More effective method based on Q-learning was introduced in [49]; wherein the resources were adjusted dynamically in cloud computing setting and performed well when posed with budget constrained over the system. Latter a more improved system based on virtual grid execution system was developed to forecast the resource consumption [50]. In order to meet with the soft deadlines for execution of scientific the job workflow HEFT algorithm was introduced in [51]. Here, it was proposed for scheduling the job workflow on IaaS setting which adds continuality for multi-tenancy in cloud computing environment [52].

3. CONCLUSION

Multi-Tenancy proves to be quite advantageous to cloud service providers. Though, it comes with the unsolved problems of accelerated computing, pipeline and effective method for scheduling. In this study, we discussed about the distinctive methods for realizingmulti-tenant frameworksand concentrated on the dense solvers over multi-core processors. We additionally talked about the architectures and pipeling utilized as a part of and the diverse ways to achieve multi-tenancy in programming paradigm. We hope the dense

evaluation and summation of the previous work will be helpful for other researchers looking for one stop study to go through the pros and cons of existing method.

REFERENCES

[1] Vaquero, L. M., Rodero-Merino, L., Caceres, J., and Lindner, M., "A break in the clouds: towards a cloud definition", *SIGCOMM Compute, Commun, Rev.,* vol. 39, no. 1, pp. 50–55, 2009.

[2] Rimal, B. P., and Choi, E., "A service-oriented taxonomical spectrum, cloudy challenges and opportunities of cloud computing", *Int. J. Commun. Syst.,* vol. 25, no. 6, pp. 796–819, 2012.

[3] Foster, I., Zhao, Y., Raicu, I., and Lu, S., "Cloud computing and grid computing 360-degree compared", *in Proc., IEEE Grid Comput. Environments Wksp,* pp. 1–10, 2008.

[4] Weissman C. D., and Bobrowski, S., "The design of the force.com multitenant internet application development platform", *in Proc., ACM SIGMOD,* pp. 889–896, 2009.

[5] Rimal, B. P., and El-Refaey, M. A., "A framework of scientific workflow management systems for multi-tenant cloud orchestration environment", *in Proc., IEEE WETICE*, pp. 88–93, 2010.

[6] Guo, Y. H. Z. H. W. C. J., Sun, W., and Gao, B., "A framework for native multi-tenancy application development and management", *in Proc., IEEE CEC/EEE,* pp. 551–558, 2007.

[7] Ballani, H., Costa, P., Karagiannis, T., and Rowstron, A., "Towards predictable datacenter networks", *in Proc., ACM SIGCOMM,* pp. 242–253, 2011.

[8] Fernandez-Baca, D., "Allocating modules to processors in a distributed system", *IEEE Trans. Software Eng.,* vol. 15, no. 11, pp. 1427–1436, 1989.

[9] Yu, J., Buyya, R., and Ramamohanarao, K., "Workflow scheduling algorithms for grid computing", *in Metaheuristics for Scheduling in Distrib. Comput. Environments,* vol. 146, pp. 173–214, 2008.

[10] Zhao, Y., Fei, X., Raicu, I., and Lu, S., "Opportunities and challenges in running scientific workflows on the cloud", *in Proc., IEEE Cyber- Enabled Distrib. Comput. and Knowledge Discovery,* pp. 455–462, 2011.

[11] Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L., and Myers, J., "Examining the challenges of scientific workflows", *Computer,* vol. 40, no. 12, pp. 24–32, 2007.

[12] Livny, J., Teonadi, H., Livny, M., and Waldor, M. K., "High throughput, kingdom-wide prediction and annotation of bacterial non-coding rnas", *PLoS ONE,* vol. 3, no. 9, p. e3197, 2008.

[13] Hsieh, F. S., and Lin, J. B., "A dynamic scheme for scheduling complex tasks in manufacturing systems based on collaboration of agents", *Applied Intelligence,* vol. 41, no. 2, pp. 366–382, Sept. 2014.

[14] Topcuoglu, H., Hariri, S., and Wu, M. Y., "Performance-effective and low-complexity task scheduling for heterogeneous computing", *IEEE Trans. Parallel and Distrib. Sys.* vol. 13, no. 3, pp. 260–274, Mar. 2002.

[15] Adulescu, A. R., and Van Gemund, A. J. C., "On the complexity of list scheduling algorithms for distributed-memory systems", *in Proc., ACM Supercomput.,* pp. 68–75, June 1999.

[16] Ristenpart, T., Tromer, E., Shacham, H., and Savage, S., "Hey,you, get off of my cloud: exploring information leakage inthird-party compute clouds", *in Computer andCommunications Security (CCS),* 2009.

[17] Gens, F., "IT cloud services user survey, pt.2: top benefits &challenges", Oct. 2008.

[18] Hagai Bar-El, "Introduction to side channel attacks", *White Paper.* Discretix Technologies Ltd.

[19] Augusto, C., "Monitoring a virtual networkinfrastructure", October 2010.

[20] Intel IT Center, "Seven Steps for Building Security in theCloud from the Ground Up ", September 2011.

[21] Mehmet, Y., Jemal, A., Tuncay, E., Andrew, B., "A Layered Security Approach for Cloud ComputingInfrastructure", *10th International Symposium on PervasiveSystems, Algorithms, and Networks* (2009).

[22] Mansfielddevine, S., "Danger in the clouds", *NetworkSecurity,* vol. 2008, no. 12, pp. 9–11, Dec. 2008.

[23] Jansen, W., and Grance, T., "Guidelines on Security andPrivacy in Public Cloud Computing", *National Institute of Standards & Technology,* 2011.

[24] Pearson, S., and Benameur, A., "Privacy, Security and TrustIssues Arising from Cloud Computing", *IEEE SecondInternational Conference on Cloud Computing Technologyand Science,* vol. 8, no. 6, pp. 692–702, Nov. 2010.

[25] Cloud Security Alliance, "Security as a Service, DefinedCategories of Service", 2011.

[26] Davida, G. I., Wells, D. L., and Kam, J. B., "Security andPrivacy", *IEEE Concurrency,* vol. 8, no. 2, pp. 24–21, 2000.

[27] Topcuoglu, H., Hariri, S., and Wu, M. Y., "Performance-effective and low-complexity task scheduling for heterogeneous computing", *IEEE Trans. Parallel and Distrib. Sys.,* vol. 13, no. 3, pp. 260–274, Mar. 2002.

[28] Blair, G., Kon, F., Cirne, W., Milojicic, D., Ramakrishnan, R., Reed, D., and Silva, D., "Perspectives on cloud computing:interviews with five leading scientists from the cloudcommunity", *Journal of Internet Services and Applications,*vol. 2, no. 1, pp. 2–9, Jun. 2011.

[29] Fard, H. M., Prodan, R., Barrionuevo, J. J. D., and Fahringer, T., "A multi-objective approach for workflow scheduling in heterogeneous environments", *in Proc., IEEE/ACM CCGrid,* pp. 300–309, May 2012.

[30] Darbha, S., and Agrawal, D. P., "Optimal scheduling algorithm for distributed-memory machines", *IEEE Trans. Parallel Distrib. Syst.,* vol. 9, no. 1, pp. 87–95, Jan. 1998.

[31] Bajaj, R., and Agrawal, D. P., "Improving scheduling of tasks in a heterogeneous environment", *IEEE Trans. Parallel and Distrib. Sys.,* vol. 15, no. 2, pp. 107–118, Feb. 2004.

[32] Gerasoulis, A., and Yang, T., "A comparison of clustering heuristics for scheduling directed acyclic graphs on multiprocessors", *J. Parallel and Distrib. Comput.,* vol. 16, no. 4, pp. 276 – 291, Dec. 1992.

[33] Liou, J. C., and Palis, M. A., "An efficient task clustering heuristic for scheduling DAGs on multiprocessors", *in Proc., Resource Management, Symp. Of Parallel and Distrib. Processing,* 1996, pp. 152–156.
[34] Bessai, K., Youcef, S., Oulamara, A., Godart, C., and Nurcan, S., "Bi-criteria workflow tasks allocation and scheduling in cloud computing environments", *in Proc., IEEE CLOUD,* pp. 638–645, 2012.
[35] He, T., Chen, S., Kim, H., Tong, L., and Lee, K. W., "Scheduling parallel tasks onto opportunistically available cloud resources", *in Proc., IEEE CLOUD,* June 2012, pp. 180–187.
[36] Xiao, Z., Song, W., and Chen, Q., "Dynamic resource allocation using virtual machines for cloud computing environment", *IEEE Trans. Parallel and Distrib. Sys.,* vol. 24, no. 6, pp. 1107–1117, June 2013.
[37] Maguluri, S. T., Srikant, R., and Ying, L., "Stochastic models of load balancing and scheduling in cloud computing clusters", *in Proc., IEEE INFOCOM,* pp. 702–710, Mar. 2012.
[38] Browning, T. R., and Yassine, A. A., "Resource-constrained multiproject scheduling: Priority rule performance revisited", *Int. Journal of Production Economics,* vol. 126, no. 2, pp. 212–228, Mar. 2010.
[39] Shue, D., Freedman, M. J., and Shaikh, A., "Performance isolation and fairness for multi-tenant cloud storage", *in Proc., USENIX OSDI,* pp. 349–362, Oct. 2012.
[40] Pandey, S., Wu, L., Guru, S., and Buyya, R., "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments", *in Proc., IEEE Advanced Information Networking and Applications,* pp. 400–407, 2010.
[41] Rodriguez, M. A., and Buyya, R., "Deadline based resource provisioningand scheduling algorithm for scientific workflows on clouds", *IEEE Trans. Cloud Comput.,* vol. 2, no. 2, pp. 222–235, 2014.
[42] Wu, Z., Liu, X., Ni, Z., Yuan, D., and Yang, Y., "A market-oriented hierarchical scheduling strategy incloud workflow systems", *J. Supercomputing,* vol. 63, no. 1, pp. 256–293, Jan. 2013.
[43] Fard, H. M., Prodan, R., and Fahringer, T., "A truthful dynamic workflow scheduling mechanism for commercial multicloud environments", *IEEE Trans Parallel and Distrib. Syst.,* vol. 24, no. 6, pp. 1203–1212, June 2013.
[44] Juve, G., Deelman, E., Vahi, K., Mehta, G., Berriman, B., Berman, B., and Maechling, P., "Scientific workflow applications on amazon ec2", *in Proc., IEEE E-Science Wksp,* pp. 59–66, 2009.
[45] Jin, J., Luo, J., Song, A., Dong, F., and Xiong, R., "Bar: An efficient data locality driven task scheduling algorithm for cloud computing", *in Proc., IEEE/ACM CCGrid,* pp. 295–304, 2011.
[46] Yuan, D., Yang, Y., Liu, X., and Chen, J., "A data placement strategy in scientific cloud workflows", *Future Gener. Comput. Syst.,* vol. 26, no. 8, pp. 1200–1214, Oct. 2010.
[47] Zhu, Q., and Agrawal, G., "Resource provisioning with budget constraints for adaptive applications in cloud environments", *IEEE Trans. Services Comput.,* vol. 5, no. 4, pp. 497–511, 2012.
[48] Ramakrishnan, L., Koelbel, C., suk Kee, Y., Wolski, R., Nurmi, D., Gannon, D., Obertelli, G., YarKhan, A., Mandal, A., Huang, T., Thyagaraja, K., and Zagorodnov, D., "Vgrads: enabling e-science workflows on grids and clouds with fault tolerance", *in Proc., IEEE High Performance Compute. Networking, Storage and Analysis,* pp. 1–12, Nov. 2009.
[49] Plankensteiner, K., and Prodan, R., "Meeting soft deadlines in scientific workflows using resubmission impact", *IEEE Trans. Parallel and Distrib. Sys.* vol. 23, no. 5, pp. 890–901, May 2012.
[50] [52] Abrishami, S., Naghibzadeh, M., and Epema, D. H. J., "Deadlineconstrained workflow scheduling algorithms for infrastructure as a service clouds," *Future Gener. Compute. Syst.,* vol. 29, no. 1, pp. 158–169, Jan. 2013.
[51] Kamesh, D. B. K., Sastry, J. K. R., Devi Anusha, C.H., Padmini, P., Siva Anjaneyulu, G., *"Building Fault Tolerance within Clouds at Network Level", International Journal of Electrical and Computer Engineering (IJECE),* vol. 6, no. 4, pp. 1560-1569, 2016.
[52] Lida Z., Qingzhong L., Lanju K., "Multi-tenant Main Memory Index Tree with Shared Structure", *TELKOMNIKA, TELKOMNIKA (Telecommunication, Computing, Electronics and Control)*
[53] vol.14, no.2A, pp. 77-84, 2016.
[54] Naresh V., Thirumala Rao, B., "A Secured Cloud Data Storage with Access Privilages ", *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)* vol. 4, no. 3, pp. 219-224, 2016.
[55] Ruoyu, W., Gail-Joon, A., Hongxin, H, and Mukesh, S., "Information flow control in cloud computing", pp. 9-12, Oct. 2010.