ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 8, August 2018

Enhanced Min-Min Task Scheduling Algorithm for Load Balancing in Cloud Data Center

Monika Verma[1], Shakshi Tiwari[2], Saurabh Sharma[3]

Research Scholar, Department of Computer Science, Gyan Ganga College of Technology, India[1]

Research Scholar, Department of Computer Science, Gyan Ganga College of Technology, India[2]

Assistant Professor, Department of Computer Science, Gyan Ganga College of Technology, India[3]

ABSTRACT: Cloud Computing is an emerging computing paradigm. It aims to share data, calculations, and service transparently over a scalable network of nodes. Since Cloud computing stores the data and disseminated resources in the open environment. So, the amount of data storage increases quickly. In the cloud storage, load balancing is a key issue. It would consume a lot of cost to maintain load information, since the system is too huge to timely disperse load. Load balancing is one of the main challenges in cloud computing which is required to distribute the dynamic workload across multiple nodes to ensure that no single node is overwhelmed. It helps in optimal utilization of resources and hence in enhancing the performance of the system. A few existing scheduling algorithms can maintain load balancing and provide better strategies through efficient job scheduling and resource allocation techniques as well. In order to gain maximum profits with optimized load balancing algorithms, it is necessary to utilize resources efficiently.

KEYWORDS: load balancing; scalability; virtual machine; time scheduling; time quantum;

I. INTRODUCTION

Cloud computing technology is composed of a large number of resources. It enables the users to access the correct and required resources in a network on demand. Applications with various resource demands require high performance computing capabilities [3]. This is based on the pay as you go approach

[1]. With the increase in demand for the computation there is a need for additional resources. Scalability improves the throughput when additional resources are added [13]. The scalability of an application is its ability to utilize the available resources without reducing the efficiency of the system. One of the ways to meet this is through load balancing. The method facilitates distribution of workload across resources and multiple computers over the network links. This helps to accomplish optimization in resource utilization, maximization of time and avoid overload of processors [8]. The services provided should be fault tolerant, highly scalable, available, flexible, less overhead, minimum cost [5]. Fundamental to this issue lays the implementation of an effective method for load balancing.

Load balancing in cloud applies specified thresholds and service level agreements (SLAs) to every request.

The distribution of tasks is one of the key challenges in cloud computing. In the existing load balancing techniques the response time is reduced when priority is not an issue. If priorities of the jobs are considered then there is no improvement in the efficiency of the response time. This paper, proposes a novel load balancing algorithm. It considers the priority of the tasks. This technique reduces the response time to process the job requests that arrives from various users of cloud. It also aims in reducing the overhead. Prioritization of the jobs is necessary in load balancing for better service to the jobs with high priority.

DOI: 10.15680/IJIRCCE.2018. 0608023

II. BACKGROUND

The study of the load balancing in the conventional distributed system is considered as the base for this algorithm.

A. Load Balancing

Load balancing is the processing of work between two or more computers, CPUs', network links, storage devices [5]. It is a method of enhancing the performance of a parallel and distributed system through redistribution of load among the processors. The reorganization of tasks among processors should maximize throughput and preserve constancy. It should also provide proper resource utilization. Load balancing is accomplished through software, hardware or both. Multiple servers are used for load balancing.

The load balancer has the following functions [3]:

1. Assigning resources for requested jobs.
2. Shifting jobs from overloaded resources.
3. Scheduling of tasks in a distributed manner.

The first one can be either system-centric or user-centric. Sometimes in order to optimize both user expectations and better resource utilization it becomes both user-centric and system-centric. The second one is more system-centric since the focus is to maintain system balance amongst the dynamical changes of the resources. System-centric algorithms do not
meet the range of the requirements given by user. User-centric algorithms do not guarantee a uniform quality of service.

The cloud environment is a highly dynamic environment with fluctuating loads and availability of resources [16]. A load balancing algorithm in high performance computing should provide services which are both system-centric and user-centric. It should refer to the attributes of resources and objectives of users, providers and system. These algorithms should provide efficient provisioning and scheduling of resources. These tasks will ensure [2]:

1. On demand availability of resources.
2. Efficient utilization of resources under high/low load.
3. Energy preservation (i.e. when usage of cloud resources is below certain threshold).
4. Reduction of cost by using less resources..

To address these requirements there is a need for a proper load balancing technique. A study of the load balancing in cloud and its objectives is required.

B. VM Load Balancer in Cloud Environment

In cloud computing environment virtual machine (VM) is the most normally used resource unit in which business services are encapsulated [6]. The important task in scheduling is to determine the storage of data and reallocation of VM. To improve the utilization of resources, virtualization is used. The energy efficiency of infrastructures relies on virtualization technologies. This can be improved by proper configuration of the VMs running in the cloud. As another key aspect, from a profit perspective, SLA compliance is also crucial [12]. Violations in SLA can result in major loss to both the client and the provider. This may also require precise and proficient SLA compliance monitoring.

DOI: 10.15680/IJIRCCE.2018. 0608023

C. Objectives of Load Balancing Algorithms

The objectives are [10]:

1. Maximize the throughput**:** A load balancing algorithm should be able to serve maximum number of requests per unit time.

2. Avoid Starvation: A request should not be in the waiting queue for an indefinite time period.

3. Minimize Overhead: Overhead causes wastage of resources. The system resources should be used properly.

4. Enforcement of priorities: High priority jobs should be given importance.

Load Balancing algorithms should satisfy the following criteria [8, 9, 14, 15]:

1. Context Switch
2. Throughput
3. Turnaround Time
4. Waiting Time
5. Response Time
6. Priority
7. Low Queuing
8. Overhead Associated
9. Fault Tolerance
10. Migration

III. ALGORITHMS

Some of the algorithms used for load balancing are discussed below [12, 8, 17, 11]:

A. Equally Spread Current Execution Load

It is a technique in which the load balancer spreads the load of the jobs among multiple virtual machines. A queue is maintained by the load balancer. It consists of the details of the tasks that are waiting to use the services and those that are currently being served by the virtual machine. Consequently the jobs are submitted to the VM manager. The job list, size and requested resources are maintained by VM manager. The job that matches the criteria for execution is selected by the balancer at the present time.

The queue and list of virtual machines is constantly examined by the balancer. The allocation of VM to that request takes place if a free VM is available to serve the node/client request. However, if there is a VM that is free and there is another VM that needs to be freed, then the balancer distributes some of the tasks of that VM to the free one. This reduces the overhead of the former VM.

The drawback of this algorithm is overhead in maintaining the list of the details of the jobs in queue. There is also the computation overhead to scan the queue again and again.

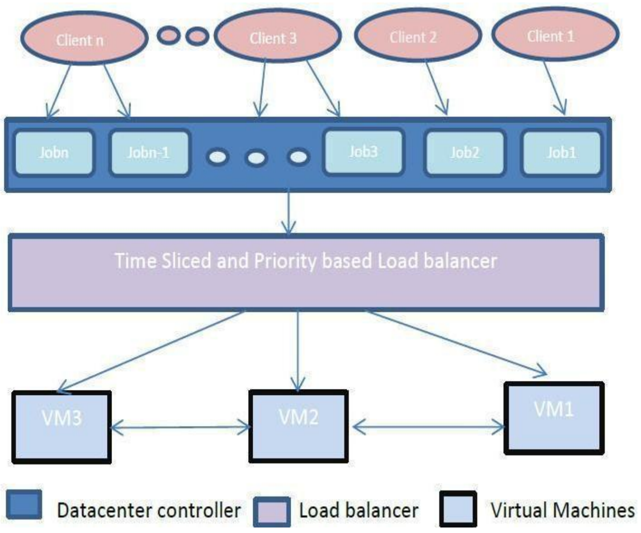
B. Round Robin

It is one of the scheduling techniques that utilize the standard of time segment. Here the time is divided into numerous segments. Each node is given a time period. The node performs its operations in this slot. The time slot is the basis for allocating resources of the service provider to the requesting client. Each user request is served by every processor within given time quantum. The execution continues for next user in the queue after the time slice is elapsed. If the user request completes within time quantum then user should not wait otherwise user have to wait for its next slot.

Context Switching is expensive in this algorithm. Context switch leads to overhead for the scheduler. In addition, it results in wastage of time and memory. There is also an additional load on the scheduler to decide the time quantum as once the job finishes its time slice it has to switch to the next task. It also has longer waiting time. Another disadvantage is low throughput.

IV. PROPOSED ALGORITHM

An algorithm "Time Sliced and Priority Based Load Balancer" for load balancing is proposed. The objectives of this algorithm are to reduce the waiting time and turnaround time. It should also prioritize the jobs with reduced time for context switching.



a time interval. Subsequently the node will perform its operations in this allotted time period. The service provider allots its resources based on time slice to the requesting client. Each processor services every user request in the stipulated time quantum. At the end of the time slice, the next queued user request comes for execution. If the user request completes its execution within the allocated time slice then it will not wait. If the time allotted was not sufficient then it has to wait for its next slot.

The next user that comes for execution will not have the same time quantum. It changes from whatever was assigned earlier. The longer the waiting time in the queue, the more time it gets for execution. The time quantum is

calculated as shown in 1.
$T = (W \div AW) * T$ 1.

Equation 2 specifies that time quantum is directly proportional to waiting time and execution time.

$Q\ \alpha\ W\ \alpha\ T$ 2

Where

W = waiting time of the client in the queue AW = average waiting time of all the clients T = the time quantum assigned to it

Q = Time Quantum

Table I gives the list of processes with burst time and priority. All these processes arrive at time 0 and use a quantum time of 9 milliseconds.

Table I Processes to be executed

Process	Burst Time	Priority
A	22	4
B	18	2
C	9	1
D	10	3
E	4	5

Fig.1. Architecture of the Proposed System

Fig.1. gives the architecture of the proposed algorithm. The load balancer receives the request from the clients. Based on their priorities it directs them to the virtual machines.

This technique utilizes the concept of time portion and priority. The time is divided into multiple slices. Time slice and priority is assigned to each node. Each node is given

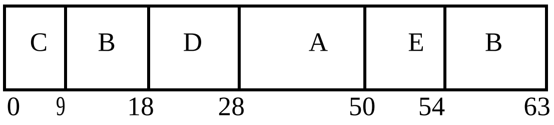


Fig. 2. Gantt chart

Using the time sliced and priority based algorithm the result is as given in the Gantt chart in Fig 2. The time quantum of each process is calculated by following the procedure that is explained in the following.

C has highest priority thus it executes first. It takes the time slice allotted to it. It is given 9 milliseconds.

B has the next priority, so its time slice is calculated based on the first formula.

W=9, AW=9, Time quantum allotted is 9 milliseconds.

$$T = \frac{9}{9} * 9 = 9$$

Similarly the time quantum is allotted for all the processes.

Average waiting time is 28. Context switching is 5. Context switching is inexpensive. The only drawback in the algorithm is the jobs with higher priority may sometimes have to wait in the queue.
The proposed algorithm is compared with round robin and equally spread current execution. It performs better in terms of turnaround time, waiting time and context switching. Table II gives the comparative results. The average waiting time, context switch and the turnaround time are measured in milliseconds (ms).

Table II Comparative Results

Algorithm	Average Waiting Time	Context Switching	Turn Around Time
Round Robin	38.2 ms	8	50.8 ms
Equally Spread Current Execution	29 ms	7	47.5 ms
Time Sliced Priority Based	28 ms	5	45.4 ms

V. EXPERIMENTAL SETUP

In order to measure the efficiency and effectiveness of Load Balancing algorithms simulation environment is required. CloudSim is a well-organized tool that can be used for modeling of cloud [11]. It is a structure developed by GRIDS laboratory which allows modeling, simulation and experimenting on designing cloud infrastructures [14].

Cloudsim framework is built on top of GridSim. The CloudAnalyst is built directly on top of CloudSim framework leveraging the features of the original framework and extending some of the capabilities of CloudSim. In the cloud lifecycle, it allows VMs to be administered by hosts. These are further taken care by datacenters. A typical cloud modeled using Cloud Analyst consists of following four units datacenters, hosts, virtual machines and application as well as system software. A basic cloud environment is set up. The load balancing algorithms can be tested here.

Datacenters has the liability of providing infrastructure services to the cloud users. Hosts in cloud are physical servers that have pre-configured processing capabilities. Host is responsible for providing software level service to the cloud users. Hosts have their own storage and memory. Processing capabilities of hosts is expressed in MIPS (million instructions per second).

Virtual machines develop and deploy customized application service models. They are mapped to a host that is a counterpart in critical features like storage, processing, memory, software and availability requirements. Application and system software are executed on virtual machine on demand. Thus, the object oriented approach of Cloudsim can be used to simulate cloud computing environment.

The traffic routing between the user base and data center is controlled by a service broker. The different policies implemented by the broker are [11]:

1. Service Proximity based routing (SPR): The service broker will route user traffic to the nearest data center in terms of transmission latency. This is based on the quickest path.

2. Performance Optimized routing (POR): Here the Service Broker actively monitors the performance of all data centers and directs traffic to the data center it estimates to give the best response time to the end user at the time it is queried.

3. Dynamically reconfiguring routing (DCR): This is an improvement to Proximity based routing. The service broker has additional responsibility of scaling the application based on the load it is facing. This is done by increasing or decreasing the number of VMs allocated in the data center.

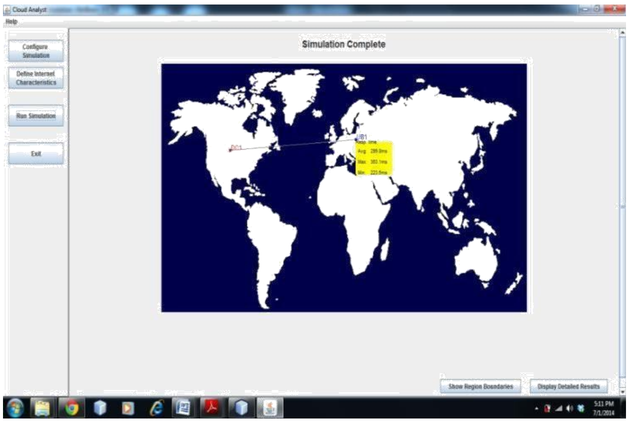
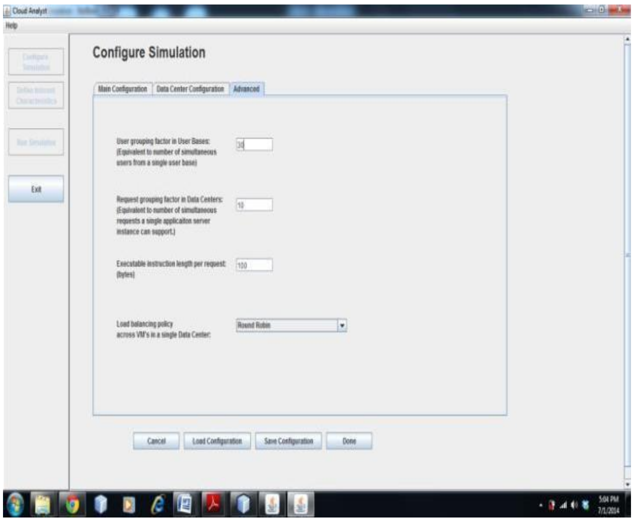


Fig.3. Cloud Analyst

Fig.3 shows the graphical user interface of the cloud analyst. Fig.4 gives the details for the configuration of the cloud analyst.

DOI: 10.15680/IJIRCCE.2018. 0608023



Configure Simulation

User grouping factor in User Bases: (Equivalent to number of simultaneous users from a single user base) 10

Request grouping factor in Data Centers: (Equivalent to number of simultaneous requests a single applicaiton server instance can support.) 10

Executable instruction length per request: (bytes) 100

Load balancing policy across VM's in a single Data Center: Round Robin

Fig.4. Configuration

VI. RESULTS AND ANALYSIS

The algorithms are run using the three different policies. The proposed algorithm gives its best performance when executed using dynamically reconfiguring routing. The results are given below.

TABLE III RESULTS OF THE ALGORITHMS

Algorithm	**Data Center Processing Time**	**Overall Response Time**
Round Robin	12.58 ms	607.62 ms
Spread Spectrum	5.55 ms	607.61 ms
TimeSliced and Priority Based	4.60 ms	607.57 ms

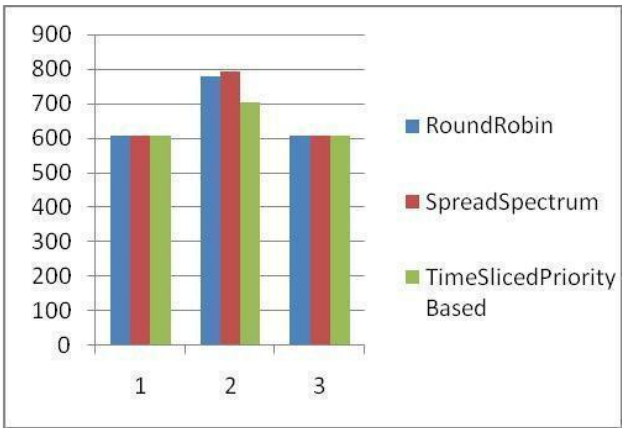


Fig.5. Response Time

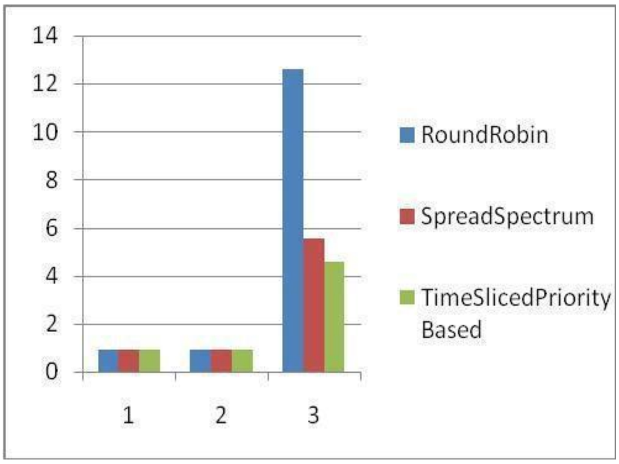


Fig.6. Data Processing Time

The x-axis in Fig.5 and Fig. 6 represents the tasks using the three different algorithms. The y-axis represents the time taken for the tasks.

TABLE IV COMPARATIVE STUDY OF THE ALGORITHMS

Algorithms	**Response Time**	**Data Center Processing Time**	**Context Switching**	**Bandwidth**
Round Robin	More	More	Expensive	Less
SS with DCR	More	Less	Expensive	Less
TSPB with DCR	Less	Lesser	Inexpensive	More

The results show that the proposed algorithm has the following advantages over the round robin and spread spectrum algorithms when used with Performance Optimized Routing and Dynamically Reconfiguring Routing.

1. Overall response time is reduced by 0.05 ms.

2. Datacenter processing also decreases on an average of 3 ms.

3. Context switching is inexpensive. On an average the number of context switch between the six processes is only five times as compared to eight and seven in round robin and equally spread current execution.

VII. CONCLUSION AND FUTURE WORK

This paper presents a novel algorithm for implementing load balancing. Load balancing is executed using time and priority. Each process is given a particular time slice/time interval based on the priority. The job with higher priority is executed first for the given time quantum. After the time slice is over, the next queued user request will come for execution. The process exits from the queue if the time slice allotted is enough to complete the task. Through this waiting time is reduced. Response time also decreases. The data center processing time is also reduced.

The time sliced and priority based algorithm is better compared to round robin and equally spread current execution load balancing algorithm with respect to waiting time and turnaround time. The context switching is also reduced. The improvement in the performance can be noted when the algorithms are run with different routing policies. The disadvantage is sometimes, the task with high priority may be the last to execute. High priority jobs at times have to wait based on the time slice. In future an algorithm should be developed which will reduce the starvation time of the jobs with higher priority.

REFERENCES

[1] Rashmi KrishnaIyengar Srinivasan, V. Suma, Vaidehi NeduMayank Katyal and Atul Mishra, **“**An Enhanced Load Balancing Technique for Efficient Load Distribution in Cloud-Based IT Industries”, Intelligent Informatics Advances in Intelligent Systems and Computing Volume 182, 2013, pp 479-485.

[2] Mayank Katyal and Atul Mishra, “A Comparative Study of Load Balancing Algorithms in Cloud Computing”, International Journal of Distributed and Cloud Computing, Vol. 1 Issue 2, December 2013.

[3] V.H. Nguyen, S. Khaddaj, A. Hoppe, Eric Oppong, “A QOS Based Load Balancing Framework for Large Scale Elastic Distributed Systems”, 10th International Symposium on Distributed Computing and Applications to Business, Engineering and Sciences, 978-0-7695-4415-1/11 $26.00 © 2011 IEEE, DOI 10.1109/DCABES.2011.12.

[4] Zhenzhong Zhang, Limin Xiao,Yuan Tao, Ji Tian,Shouxin Wang,Hua Liu, “A Model Based Load-Balancing Method in IaaS Cloud”, 42nd International Conference on Parallel Processing, 0190-3918/13 $26.00 © 2013 IEEE, DOI 10.1109/ICPP.2013.95.

[5] Chandrasekaran K. and Usha Divakarla, “Load Balancing of Virtual Machine Resources in Cloud Using Genetic Algorithm”, ICCN 2013, pp. 156–168, Elsevier Publications 2013.

[6] Wubin Li, “Algorithms and Systems for Virtual Machine Scheduling in Cloud Infrastructures”, Ph.D Thesis, April 2014, Dept. of Computing Science, Umea University, Sweden.

[7] Nada M. Al Sallami, “Load Balancing in Green Cloud Computation”,Proceedings of the World Congress on Engineering 2013 Vol II, WCE 2013, July 3 - 5, 2013, London, U.K., ISBN: 978-988-19252-8-2, ISSN:2078-0958 (Print); ISSN: 2078-0966 (Online).

[8] Soumya Ray and Ajanta De Sarkar, “Execution Analysis Of Load Balancing Algorithms In Cloud Computing Environment”, International Journal on Cloud Computing: Services and Architecture (IJCCSA), Vol.2, No.5, October 2012, DOI: 10.5121/ijccsa.2012.2501.

[9] Rizwan Maredia, “Automated Application Profiling and Cache Aware Load Distribution in Multi-Tier Architecture”, Thesis, School of Computer Science, McGill University, Montreal, 2011. N. Meghanathan and G. W. Skelton, “Risk Notification Message Dissemination Protocol for Energy Efficient Broadcast in Vehicular Ad hoc Networks,” *IAENG International Journal of Computer Science*, vol. 37, no. 1, pp. 1–10, Jul. 2010.

[10] Ishwari Singh Rajput and Deepa Gupta, “A Priority Based Round Robin CPU Scheduling Algorithm for Real Time Systems”, IJIET, Vol:1, Issue 3, ISSN: 2319-1058, October 2012.

[11] Bhathiya Wickremasinghe and Dr. Rajkumar Buyya, “CloudAnalyst::A CloudSim-based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments”, MEDC Project Report, 22/6/2009.

[12] Ankita Sharma and Upindar Pal Singh, “Energy Efficiency in Cloud Data Centers using Load Balancing”, International Journal of Computer Trends and Technology(IJCTT), Vol.11, No.4, ISSN:2231-2803, May2014.

[13] Guilherme Galante and Luis Carlos E. de Bona, “A Survey on Cloud Computing Elasticity”, UCC, 2012, Utility and Cloud Computing, IEEE Internatonal Conference on, Utility and Cloud Computing, IEEE Internatonal Conference on 2012, pp. 263-270, doi:10.1109/UCC.2012.30

[14] Akanksha Chandola Anthwal and Dr. Nipur, “Survey of Fault Tolerance Policy for Load Balancing Scheme in Distributed Computing”, International Journal of Computer Applications(0975-8887), Vol.74, No.15, July 2013.

[15] K.C. Ofakar, Ugwoke F.N, Okezie C.C, “Gateway Load Balancing Service in Cloud Data Center Environments using Throughput Metric Index”, American Journal of Computation, Communication and Control, 2014, 1(1), 8-17.

[16] Kyoungho An, Shashank Shekhar, Faruk Caglar, Aniruddha Gokhale and Shivakumar Sastry, “A Cloud Middleware for Assuring Performance and High Availability of Soft Real-Time Applications”, Elsevier Journal of System Architecture, Feb 2014.

[17] Tanveer Ahmed and Yogender Singh, “Analytic Study of Load Balancing Techniques Using Tool Cloud Analyst”, International Journal of Engineering Research and Applications (IJERA), ISSN: 2248-9622, Vol. 2, Issue 2, April-2012, pp. 1027-1030.