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

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 3, March 2014

PSO Based Approach in a Hybrid Gaming Environment for Effective Load Distribution in Cloud Computing

Rajesh George Rajan[1], V.Jeyakrishnan[2]

M.Tech. Student, Department of Computer Science, Karunya University, Coimbatore, India[1]

Assistant Professor, Department of Computer Science, Karunya University, Coimbatore, India[2]

Abstract: Online gaming is one of the major sectors in the business environment. Through the internet and the networks these types of games became very popular in the society. Considering this gaming in cloud environment with the cloud and peer nodes, greedy strategies is used to set up a better utilization among these nodes. But by using the greedy approaches the execution time of the corresponding system will increase. In this paper we propose a better optimization technique called particle swarm optimization (PSO). The major role is to reduce the execution time of the system. So this work proposes a hybrid distributed gaming environment for an effective utilization of the cloud and peer nodes and also to reduce the execution time of this execution time.

Keywords: Cloud computing, Load distribution, Distributed virtual environments, Online gaming.

I. INTRODUCTION

Cloud computing is one of the most important part of the current computational environment. Cloud computing has computational and also have some sociological concerns. Cloud computing is on demand service by which can be provide various types of services to our societies. There were various types of services that should be provided by the cloud computing they are platform as a service, infrastructure as a service and software as a service. Cloud computing was intended to enable computing across widespread diverse resources rather than on local machines or at remote service farms. Although there is no proper definition for cloud computing. Load balancing was identified as a major concern to allow cloud computing to scale up to increasing demand. Load balancing is the process of reassigning the total loads to the individual nodes of the collective system to make the best response time and also good utilization of the resources.

Cloud Computing [1] allows us to solve the aforementioned scalability and hardware ownership problems because of on demand resource provisioning [2, 3]. The possibility of renting machines lifts the DVE operators from the burden of buying and maintaining hardware, whereas it offers the illusion of infinite machines, with good effects on scalability. Also, the pay-per-use model adheres to the seasonal access pattern of the DVE (e.g. more users in weekends than in the middle of the week). However, Cloud Computing may still be costly for platform operators. Besides server time, bandwidth cost represents a major expense when operating a DVE [4]. When this cloud approaches is to be very feasible but its cost must be too higher for the distributed virtual environments. So that another concept of infrastructure for the distributed virtual environments is considered that is peer-to-peer concept. So various advantages are to be there for the peer system that is the network is able to self repair, robustness and also the major thing is the low cost that can be affordable to the organization. So these are the two orthogonal approaches that are to be combined.

When considered the current commercial market, the massively multiplayer online games rely a centralized architecture. This centralized architecture supports various functionalities like identification, management of virtual environments etc. In the massively multiplayer games, the major difference compared to other online games was the players who share the same virtual environment, even if more amount of players in the games. So the major identity is the virtual representations of the objects, called avatars. These avatars are shows the state in the virtual environments.

In the gaming environment, the node pool contains cloud and peer nodes. So the peer-based system ignores various problems like bootstrapping, system security etc. Peers and clouds are worked together in order to distribute the workload and also allowing to prevent the fault occurrence in the nodes. This will also helps to reduce the cost of the system and also for a better efficiency. Massively multiplayer online games are forced to overprovision the resources of their architecture to maintain the load. There were huge numbers of users are to be found in peak and non peak times. So this is the need of cloud computing in the gaming environment.

The main purpose of the massively multiplayer online game operators are they utilize cloud resources at during the peak time and release them when they are not used. So this massively multiplayer online games are use this cloud as the infrastructure as a service. The paper is structured as follows. Section 2 describes the related work, section 3 describes hybrid architecture, and section 4 describes results and finally describes the conclusion.

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

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 3, March 2014

II. RELATED WORK

Various load distribution and different hybrid architectures are to be already found. So in this work we take different architectures and load distribution policies. When considering the hybrid architectures for the distributed virtual environments, the main aim is to utilize the combination of peers and the centralized servers. Hybrid architectures are mainly classifies according to two different approaches. The first approach is the corresponding region can be assigned either to a peer or to a server. And another approach is a group of cells can be assigned to peers.

According to [5] these authors proposed the method as in the first category. Considering the square cells and that was managed by the central server. In this the higher bandwidth and computational capabilities to enter in to a cell becomes the cell manager. And some cells act as the backup managers to increase the failure robustness. In [6] the authors proposed an approach as in the second category. Here a central server becomes a major part and the other peers are run like other game genres. Here the separation of the distributed virtual environments as certain instances, so that there were a fixed amount of players are shared the distributed virtual environments.

Considering the section in [7] the similar way can be seen. Here the partition can be takes place according to the functions that can be done by the servers. Servers can do some functionalities like authentication, persistence etc. So by using these functionalities the partition can takes place. The authors of [8] provide the distinction between the important parts of the gaming environments. That is the state changing actions and position changing actions. These authors proposed a hybrid architecture having peers and the servers. The peers manage the positional changes and the servers manage the state changes. These changes are to be updated to the distributed hash tables so that at each time the state or a position of the corresponding nodes will change it will make some changes in the distributed hash table. When considering the load distribution in the distributed virtual environments the virtual representation called avatars becoming the important part. These avatars are move across the distributed virtual environments and communicate with each other. During these time the state of the avatars become changing. There were direct and indirect interactions are to be done by the avatars in terms of computational power and the band width. Load balancing is the major problem that can be faced by the current environments. In [9,10] the virtual environments are divide in to small cells called the micro cells and by using the cluster of micro cells by the servers the load distribution can takes place. The comparison takes place between the adjacent microcells is the crucial point in the load distribution so that the highest one can distribute the load towards the lowest one. According to the authors [11], the load migration can be done according to the heat diffusion mechanism. According to the heat diffusion algorithm, the virtual environment is divided in to large number of square cells and each square cell having objects. The working of the heat diffusion algorithm is in such a way that every node in the cell sends load to its neighboring nodes in every iteration and the transfer was the difference between the current node to that of neighboring node. So it was related to heat diffusion process. That is the transfer of heat from high to low object, when they were placed adjacently.

When considering the load distribution according to the work [12] the partition can takes place according the partition of the virtual environment surface and then assigns a server to each of the voronoi cell. According to their capacity the servers are moves on the surfaces. So based on the capacities the load can be moves from one place to another. In [13] each server is defined by the cluster of players. And in these case if the server cannot withstand more number of clusters then some of the players in the cluster will move to the another cluster or the entire cluster will move from one server to another.

III. HYBRID DVE ARCHITECTURE

Hybrid architectures are used to exploit and combine the various user resources i.e. the peer and the servers that is the cloud. According to this section the overall structure of the distributed virtual environment are to be discussed. In the gaming environments there were various players and each players having their own states and their own position in order to maintain the proper functioning of the game. The players are to be connected to the server by means of the game client and the client show the representation of the corresponding game. When each position is to be updated by the client then the positional action manager can be updated. Similarly the state action manager can also update the position of the state. i.e. the main work of the state action manager is the updating of the state of each player in the game. So if a player died then the state of that player can be updated in the state action manager. Currently there were distributed architectures having the multiple nodes distribution and having multiple servers. There were various elements are to be come under this architecture. So the overall architecture becomes more complex one. The virtual environments are divided or clustered according to their spatial position. So the area of interest of the players can be partitioned. Here the load can be increasing in the region of high amount of players are to be situated. At the same time the different positions of the game can be updated on the positional action manager. In this situation the interactiveness of the gaming environments become reduced.

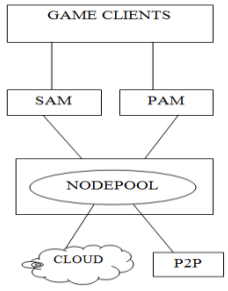

Fig.1 Overall Architecture (Source Elsevier Computer System29, pp.1564)

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

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 3, March 2014

Fig.1 presents the major components of the architecture. The state action manager provides the infrastructure of the managements of the state actions. Here the combination of the cloud and peer nodes are to be arranged in a beautiful manner. So the performance of the cloud and peer nodes are to become the major factor of the current distributed gaming environments. In the state action manager, there were clients are to be there. In this client the connection between the servers can be established and shows the virtual representation can be done to the players. At the same time there were servers can be there. It manages the state actions coming from the clients and also manages the various requests coming from the different clients.

There were also a backup server can be there in order to maintain when a problem occur in any of the server, then the backup virtual server can act as the server. So in this environments the cloud node can be choose as the backup virtual server. The core components of the architecture store both the avatars, which are the virtual representations of the users in the virtual world, and the passive objects which are not controlled by players, and are characterized by a state which is shared and modifiable by the avatars. [14**]** Any entity, both avatars and passive objects, is stored in both the components of the system. The position of the entity is stored by the PAM so that the load on the PAM is mostly due to the positional actions of the entities. The rest of the state of the objects, for instance the level of energy of an avatar or the state of a door, is instead stored by the SAM. The load on the SAM is generated by direct interactions between avatars or by indirect interactions between avatars due to avatars interaction with passive objects.

According to the load distribution strategy there were various approaches are to be there. Here the greedy approaches are to be used for the better utilization of the nodes and also for the better load distribution. In this case the greedy heuristic with state is the approach to get the better utilization of the nodes. Comparing the greedy approaches we see that the selection of the best node from the node pool can be done in this approach. When considering the greedy heuristic with state there were a time bound to the selected node. So that time only that node can be participated in the load distribution. So that every node in the node pool can be participated in the load distribution and as a result the utilization of the node can becomes higher.

Greedy Heuristic with State: This is similar to that of greedy heuristic. But the difference is the introduction of the state. That is there were a time was to be set for the selection and performance of a node. The algorithm is same as that of the greedy heuristics. The importance is that the continuous usage of the single node was cannot be allowed here. The pseudo code of this approach is as follows

1. Initially set the node pool becomes null
2. For each updating of cloud and peer nodes in to the pool calculate the score of the nodes
3. According to the basis of the score, nodes are arranged in descending order
4. Selection of the best node from the pool by comparing the load of the node
5. Load distribute through the best node
6. Update the load of the best node
7. Set a minimum interval of time for the selection node in order to avoid continuous selection
8. Repeat the step 4 to 6

So this is about the greedy heuristic with state approach. But when considering the execution time of the corresponding approaches make very high in the gaming environments. Here the selection of the nodes can be done from the node pool by comparing the node load of each of the node. So for that process lot of time can be taken for the comparison and the selection. So the improvement of the execution time by using the particle swarms optimization method.

Particle Swarm Optimization: Optimization is the integral part of our current life. Optimization problems arise in various disciplines like manufacturing systems, recognition patterns etc. Particle swarm optimization is a simple and robust based on the social and cooperative behavior shown by various species like flock of bird etc. Here we use this particle swarm optimization logic to provide a better execution time for the environments. The pseudo code of this approach as follows.

1. Initialize the position and velocity randomly.
2. Initialize the node weight. Calculate score value for each particle.
3. Calculate pbest and gbest for each particle
4. Update the position and velocity of each particles
5. Update pbest for each particle if its current value is better.
6. Update gbest for each particle
7. Update the node weight of the processed node.

IV RESULTS

When considering these two approaches in fig.2 shows the execution time of the greedy heuristic with state approach. According to this graph the execution time taken by each of the node in that process can be shown below. So each node can takes some amount of time for the execution. After that it can be goes to the other node.

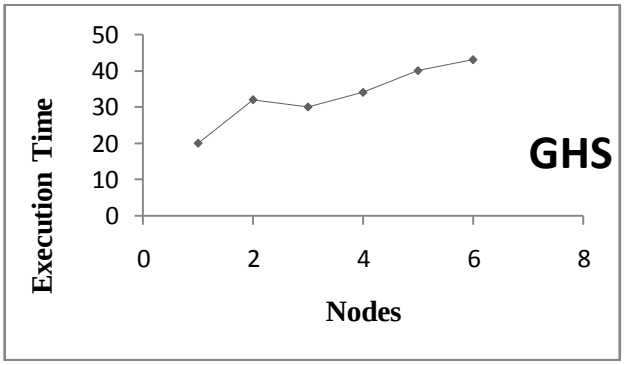

Fig.2 Execution time in GHS

Considering the execution time of the particle swarm optimization, the execution time shows a better one. Here

the execution time of this compared to the greedy heuristic approach the execution time of the particle swarm optimization is the best one. In the particle swarm optimization the selection of the node can be done at random so the value of the node can be compared with only the neighboring nodes so that comparison of all the nodes in the node pool cannot be checked out.

So at each time the position and the velocity of the node can become changed. In this particle swarm optimization the selection of node can be takes place in random. And the overall comparison cannot be done. So as shown in the below fig.2 we see that at each time the node can be selected and within minimum amount of time the execution of the node will happen.

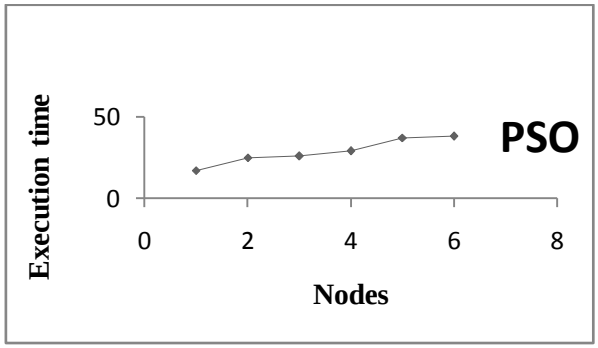

Fig.3 Execution time in PSO

In this so we see that the execution time of the particle swarm optimization is reducing or decreasing.

V. CONCLUSION

In this paper we consider the greedy strategy like greedy heuristic with state for the better utilization. When considering the gaming environment the effective execution time of the nodes are become the one of the challenge. This can be reduced by the help of the optimization method called particle swarm optimization .In this paper the particle swarm optimization is the better algorithm for the reduction of execution time of nodes in the gaming environments.

REFERENCES

[1] R. Buyya, C. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility, Future Generation Computer Systems 25 (6) (2009) 599–616.

[2] V. Nae, A. Iosup, S. Podlipnig, R. Prodan, D. Epema, T. Fahringer, Efficient management of data center resources for Massively Multiplayer Online Games, 2008 SC—International Conference for High Performance Computing, Networking, Storage and Analysis, 2008, pp. 1–12.

[3] V. Nae, R. Prodan, A. Iosup, T. Fahringer, A new business model for massively multiplayer online games, in: Proceeding of the Second Joint WOSP/SIPEW International Conference on Performance Engineering, ACM, 2011, pp. 271–282.

[4] K. Chen, P. Huang, C. Lei, Game traffic analysis: an MMORPG perspective,Computer Networks 50 (16) (2006) 3002–3023.

[5] K. Kim, I. Yeom, J. Lee, HYMS: a hybrid mmog server architecture, IEICE Transactions on Information and Systems E87 (2004) 2706–2713.

[6] I. Barri, F. Gine, C. Roig, A hybrid P2P system to support MMORPG playability,in: 2011 IEEE International Conference on High Performance Computing and Communications, 2011, pp. 569–574.

[7] A. Chen, R. Muntz, Peer clustering: a hybrid approach to distributed virtual environments, in: Proceedings of 5th ACM SIGCOMM Workshop on Network and System Support for Games, ACM, 2006, p. 11.

[8] J. Jardine, D. Zappala, A hybrid architecture for massively multiplayer online games, in: Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games, ACM, 2008, p. 60.

[9] B. De Vleeschauwer, B. Van Den Bossche, T. Verdickt, F. De Turck, B. Dhoedt, P. Demeester, Dynamic microcell assignment for massively multiplayer online gaming, in: Proceedings of 4th ACM SIGCOMM Workshop on Network and System Support for Games, ACM, 2005, pp. 1–7.

[10] D. Ahmed, S. Shirmohammadi, A microcell oriented load balancing model for collaborative virtual environments, in: Virtual Environments, Human- Computer Interfaces and Measurement Systems, 2008, VECIMS 2008, IEEE Conference on, no. July, IEEE, 2008, pp. 86–91.

[11] Y. Deng, R. Lau, Heat diffusion based dynamic load balancing for distributed virtual environments, in: Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology, ACM, 2010, pp. 203–210.

[12] M. Esch, E. Tobias, Decentralized scale-free network construction and load balancing in Massive Multiuser Virtual Environments, in: Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom, 2010, 6th International Conference on, IEEE, 2010, pp. 1–10.

[13] S. Rieche, K. Wehrle, M. Fouquet, H. Niedermayer, T. Teifel, G. Carle, Clustering players for load balancing in virtual worlds, International Journal of Advanced Media and Communication 2 (4) (2008) 351–363.

[14] Emanuele Carlini, Laura Ricci, Massimo Coppola, Flexible load distribution for hybrid distributed virtual environments: Future Generation Computer Systems 29 (2013) 1561–1572.