

Efficient Distribution of Virtual Machines using Honey Bee Windowing in Cloud Computing

Firos A

Department of Computer Science & Engineering,
Rajiv Gandhi University
Doimukh, Arunachal Pradesh, India
firos.a@rgu.ac.in

Seema Khanum

Department of Computer Science & Engineering,
Rajiv Gandhi University
Doimukh, Arunachal Pradesh, India
seema.khanum@rgu.ac.in

Abstract—This paper presents an approach for developing virtual computing layer which is used by the cloud terminals during the virtual infrastructure downloading. The virtual computing layer is cable of image creation, maintenance and downloading on behalf of the cloud terminals. Using such virtual computing layer will considerably reduce the load maintained between the virtual computing client and the server. It also schedules the image downloading in the cloud infrastructure based on software image windows. The Honey bee foraging algorithm performs the work in different phases while creation of virtual images. The forage bees will search for the suitable image portion in the neighborhood and the scout bees will bring the found portion from the location indicated by the forage bees.

Keywords— *Cloud Computing; Distributed Virtual Environments; Honey Bee Algorithm; Windowing*

I. INTRODUCTION

In recent years, the advance in virtualization is being driving the evolution of networking technologies. Even though the performance issues of the virtualized software are a matter of concern, virtualization technology is widely accepted by a large mass. Emergence of huge bandwidth networking technology has posed new challenges for developing high performance virtualized software. The major challenge in virtualized client-server interaction is its performance degradation, due to packet loss from a remote server. This major challenge can be addressed by introducing dedicated cloud terminals between virtual client and server.

In realistic environments, there exist more complicated application scenarios, such as migrating VM due to re-source preemption, which may lead to dynamic changes in reliability parameters of the host server[1]. This paper proposes scalable client-server interaction software wherein, the client interacts with its nearby cloud for downloading its virtual computing environment. The objective evaluation of the results shows that, the proposed technique provides high robustness against packet loss and also achieves a better performance while downloading larger software for the client even when the interaction with the server owe indirectly with a nearby cloud terminal.

This paper is organized as follows. Section II describes the overview of scalable client-Server virtualized environment and about operating system downloading from a server

through cloud terminal. Section III describes an efficient Honey Bee Windowing algorithm that significantly improves the performance of the virtualized client-server interaction. Finally, Section IV concludes the paper.

II. SCALABLE CLIENT-SERVER VIRTUALIZATION

A. Client Server Virtualization

The I/O facility in existing method of client server virtualization involves the use of I/O operation of TCP for individual files when an operating system is to be downloaded from the server to the client. This creates a bottleneck for the client, mainly when the server sits apart and also if there is fairly high packet loss in the interaction. To avoid such a performance bottleneck, the advantages of cloud computing will play an important role. But, the cloud computing terminal usually works as a facilitator for the client, where in the service requested by client will be given from its perimeter. This again may degrade the performance of client-server interaction. Hence, the optimal solution for performance degradation can be provided by upgrading cloud computing device from a mute downloader to an organizer. The main tasks of the organizer is to download the files for the first request from its terminal and then to organize these files into a single image. Once the image generation is done, depending on the criteria of the client request, the same will be passed to the client. Upon receiving, the client will extract required files from this image file for its overall activity.

B. Window Based Mechanism

Cloud message windowing will be done in order to reduce the delay for the first request from the client. Downloading individual files for both the computing environment and the operating system and then making it as an image is an extremely time consuming task. So, the image creation will be divided into a group of sub activities for making a full unified image – that is a window of n slots will be created in the cloud terminal [2] where in the slots will be filled either from the VC server or from the nearby cloud that has the same (required) slot. So when the terminal first contacts the nearby cloud it simultaneously sends the request to the other nearby clouds and to the VC server. Since downloading and making sub image from VC server is a time consuming process, it will give the preference to the Virtual Terminal (VT) neighbor.

Thus, the image creation in VT will be faster and may be given to the client.

C. Brief on Cloud Computing

Nowadays, cloud computing is used for solving many problems of wider scope and greater depth. It has given new avenues for performance as several companies and products finds newer ways to cut cost and improve productivity in the process. 82% of the companies reportedly saved money by moving to the cloud. According to research by computer weekly publisher, TechTarget, around 53% of the organizations worldwide are now using some or the other form of cloud services. In near future cloud may become the default delivery method for new IT products. An outline of a cloud infrastructure is given in Fig.1.

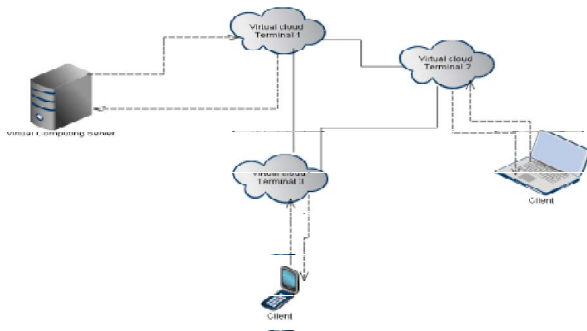


Fig. 1. A cloud infrastructure

There is an estimated one Exabyte of data stored in the cloud and its global data center traffic is higher than three Zettabytes [4].

D. Virtual Machine

Virtual machine is the emulating software, which emulates a physical computing environment where in an operating system or a program can be installed and executed. Usually, virtual machines are deployed within a virtualization layer that runs over other operating system (OS), generally called as the host OS. Thus, many individual and independent virtual computing environments (i.e., guest OS) can be deployed on top of the host OS using a single virtualization layer. The major advantage of VM is that it ensures applications and services that run within VM cannot interfere with the host OS or the other VMs.

The virtualization layer typically manages the requests arising from the VMs for resources, such as CPU, memory, network and other hardware resources, by translating these requests to the underlying physical hardware. For optimizing hardware resource utilization, VMs are usually moved, copied and reassigned between host servers [5].

Since each virtual machine though cloud exists in its own space that's not tied down to data center, it simplifies the

entire architecture when comes to performance measure of that virtual machine.

E. The Study and Cause of Study

This study is based on VMX server and VMware cloud foundry, which is already gaining attention from various open source and cloud companies. The random arrival of load in a cloud server environment can cause some server to be heavily loaded while other server is idle or only lightly loaded [3]. So, the experts believe that cloud foundry has not offered robust service hence had shown two serious outages recently. The first outage was the intermittent failure in the Droplet Execution Agents, which is considered as the serious outage to be solved productively, otherwise the failure may spread to the other nodes as well and the total infrastructure may collapse. Second more significant outage was the cut off customers' access to their applications and network all together.

Ironically, the second outage was a result of human error in an effort to avoid outages such as the first one on the longer run, which only blocked customer's access to the backend of their applications. But the first one was never a direct human error but it was due to the elevated asymmetric load in Droplet Execution agents. A solution to the above issue is proposed in this paper through Honey Bee Windowing algorithm.

III. HONEY BEE WINDOWING ALGORITHM

This algorithm is derived from honeybee foraging algorithm. It adapts the behavioral of honey bees for finding and bringing food [4]. This algorithm is based on the behavior of honey bee which forage (search widely for food). When the honey bee finds one location of food, it comes back to the beehive to inform this using a waggle dance. This display of dance gives the idea of the quantity of food and also its distance from the beehive. Scout bees then follow the forage to the location of food and reap it.

The proposed algorithm is similar to the behavior of forage and scout bees while it makes widow for the image to be given to the client by the cloud. Overall activity of software image creation in different nodes of system is depicted in Fig.2.

Step 1: In this resource allocation model a Server Colony (SC) will be maintained, which is the service provider for the Virtual Software Infrastructure (VSI) of the clients. Clients are customers which sends a job request say x , indirectly through the Cloud Infrastructure Terminal (CIT_p) maintained between VITs and SC. As explained in the Section II, here the cloud terminal is no more a mere downloader, but it measures the feasibility of fetching the job requested by the client to its entirety as fast as possible. For this the virtual terminal first checks in its heap whether the job request's window is available locally or not. If it is unavailable a 'read image' request on behalf of the client will be given directly to the Server Colony (SC). Now the SC will give back the response based on the condition –the question arises here is, is it the first request for the Virtual Software Infrastructure (VSI)? If

the answer is 'YES', SC will make the VSli and its corresponding Slot Advertisement Window for i(SAWi). Each entry of SAWi will contain a vector <Slot number, owner cloud(s)>, Where each slot size will be a couple of pages (fixed). Now slot one with SAWi will be given to CITp. So, initially SAWi's first entry will be <1, p>. On the other hand if the answer is 'NO', SC will fetch the slot (x, SAWi) and will be given to CITp. Server on demand software image creation process is depicted in Fig.3.

In CITp, SAWi will be given to its Wall of Advertisement (WoAi). WoA may contain information about all VSIs loaded in SC. From WOAx temporary slot Advertisement Window (tSAWi) will be created and based on this a window formally created in the heap of CITp. Parallel to this operation step-2 operation will be done by CITp.

Step 2: CITp forage bees will pass to other clouds, for the discovery of remaining slots. Upon receiving the request from step-2 of CITp, a CIT say q (CITq), will give to the discovery packet its information, if it contains any of the slots of VSli

investigated by the forage bee from VITp. Now the bee will go back with information and do a waggle at beehive (VITp).

Step 3: On waggle dance of bees CITp updates its WOAp and tSAWi will be generated.

Step 4: Being tSAWi with probable owners of the slot, the scout bees will go to that location and begins to reap. If the slot is not presently available with CIT, bee will request the present Cit for the next probable CIT having the slot and goes to that CIT and so on. Finally the scout bee will go back with slot once it gets its food otherwise when it visits a predefined number of CITs (say 5).

Step 5: Scout bees comes back with slot from where it got the slot a tuple (slot number, location). Now, window filling at location VITp will be initiated after a predefined time tp (say average time to visit a maximum of CITs in sequence). In the meantime at regular intervals r, that is at r, 2r, 3r... tp steps 2, 3 and 4 will be repeated. By the time tp, CITp expects all the scout bees to come back and fills the window with slots it reaped.

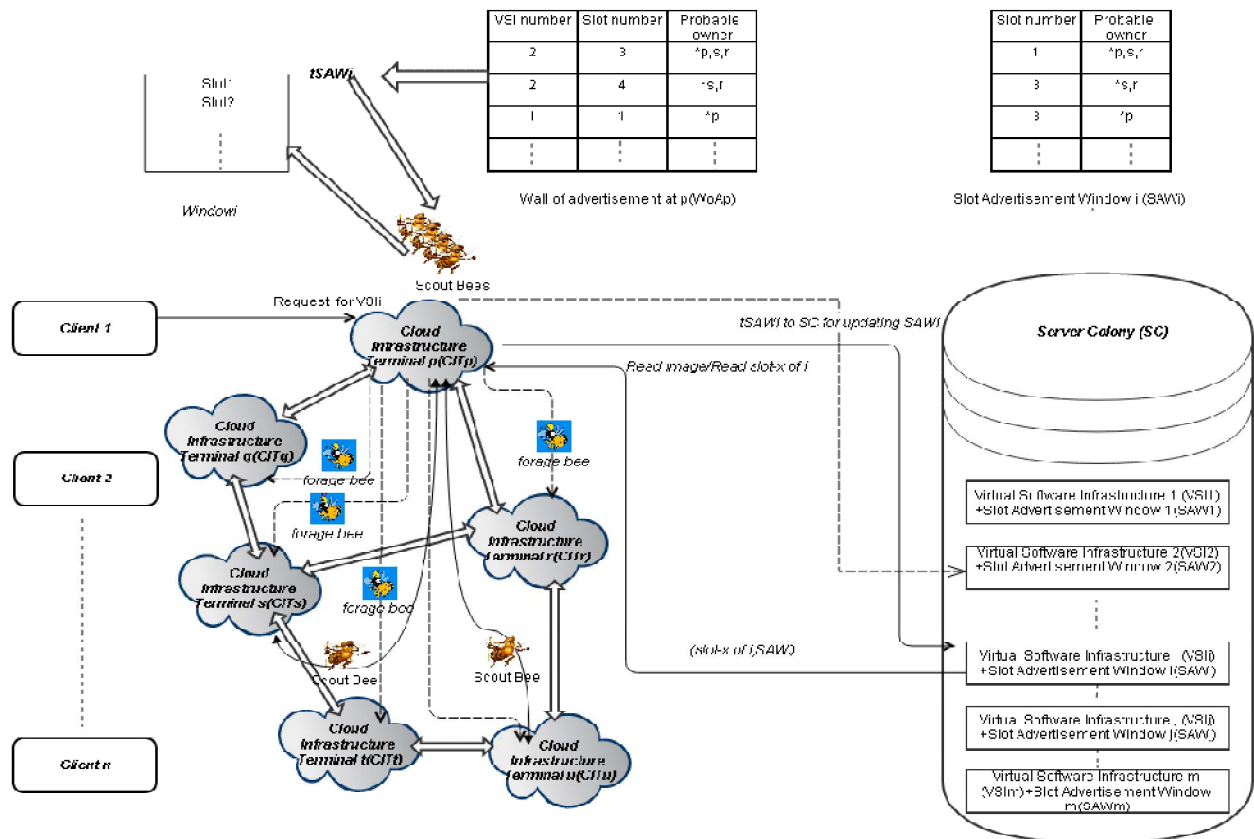


Fig. 2. Honey bee windowing process

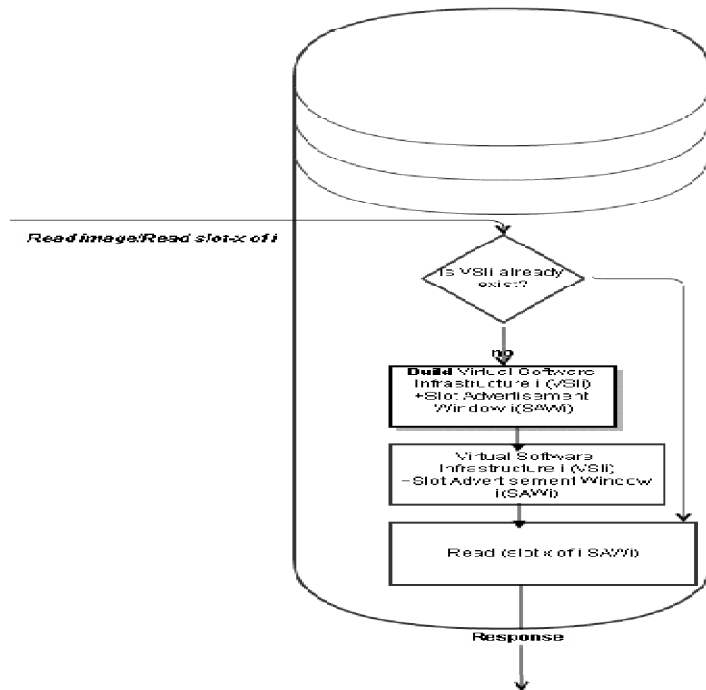


Fig. 3. Server on demand software image creation process

IV. CONCLUSION

In this paper, an extensive load balancing approach is presented for cloud computing which can be used in virtual computing infrastructure. The study considered the loads of both virtual computation server as well as the cloud computing server, in maintenance as well as the data processing and data storage.

The asynchronous elevated load in the cloud terminals is a matter of concern when it deals with large 'software as service' infrastructure, like virtual computing infrastructure. For typical Cloud Computing Infrastructure, downloading may consume comparatively more load than Virtual Storage Infrastructure. So, the cloud infrastructure deals with Virtual computing devices must have better strategy of downloading, maintenance and processing the image which client gets from the server.

This paper proposes a mechanism where the load in cloud terminals may get decreased when image creation is to be divided into a group of sub activities, which in turn will help in making a unified full image. This can be accomplished using windowing strategy where in filling of the windows may be done though the efficient algorithm of honey bee forage.

The conclusion is that the asynchronous increasing load in the cloud computing terminal may be reduced when it incorporates the windowing strategy for image creation and honey bee foraging technique for downloading the image in scheduled manner. Using this approach, it is found that cloud computing service can provide better performance especially when the clients' requests are highly computational and frequent with the server.

REFERENCES

- [1] Matthias Schmidt, Niels Fallenbeck, Matthew Smith, Bernd Freisleben, "Correlation Modeling and Resource Optimization for Cloud Service with Fault Recovery", IEEE Transactions On Cloud Computing, IEEE Computer Society, , 2168-7161 (c) 2016 IEEE
- [2] Jiyin Li, Meikang Qiu, Jain-Wei Niu, YuChen and Zhong Ming, "Adaptive Resource Allocation for Preemptable Jobs in Cloud Systems," IEEE International Conference on Intelligent Systems Design and Applications, pp. 31-36, 2010.
- [3] Jaspreet kaur "Comparison of load balancing algorithms in a Cloud", International Journal of Engineering Research and Applications (IJERA), ISSN: 2248-9622, Vol. 2, Issue 3, May-Jun 2012, pp.1169-1173.
- [4] Dhinesh Babu L.D, P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments", Applied Soft Computing 13 (2013), pp.2292-2303.
- [5] Matthias Schmidt, Niels Fallenbeck, Matthew Smith, Bernd Freisleben, "Efficient Distribution of Virtual Machines for Cloud Computing", Proceedings of the 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing, IEEE Computer Society Washington, DC, (2010), pp.567-574