National College of Ireland

Efficient virtual machine allocation and optimization of response time in cloud delivery network

MSc Research Project
Cloud Computing

Deep Shah
Student ID: X17141702

School of Computing
National College of Ireland

Supervisor: Victor Del Rosal

National College of Ireland

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	**Deep Shah**
Student ID:	**X17141702**
Programme:	**MSc. Cloud Computing** **Year: 2019-2020**
Module:	**MSc. Research Project**
Lecturer:	**Victor Del Rosal**
Submission Due Date:	**18/04/2019**
Project Title:	**Efficient virtual machine allocation and optimization of response time in cloud delivery network**
Word Count:	**6127** **Page Count: 22**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: **.....** **.....**

Date: **18/04/2019**

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Efficient virtual machine allocation and optimization of response time in cloud delivery network

Deep Shah
X17141702

Abstract

Cloud Computing is one of the most widely used domain by various researchers for testing and simulating their theory and research projects. The three main cloud services are Infrastructure-as-a-Service (IaaS), Platform-as-a-service (PaaS) and Software-as-a-Service (SaaS). But with the recent growth of cloud there are many more new ways in which the different types of applications can be developed and deployed by the means of virtualization, along with how the different services can be provided to the end users. This growing advent has the introduction to Cloud Delivery Network, which is different than the traditional virtual Content Delivery Network. This research project proposes a novel framework for Cloud Delivery Networks (CDNs). The novel framework based on CDNs provides the cloud content delivery services as customized and required by the customers/users. The cloud can be a public, private or hybrid. The aim is to share the virtual machines (VMs) in Infrastructure-as-a-Service in cloud-based CDNs, keeping the SLA intact. This framework also allows the VM capabilities for scale up and scale down as per the dynamic changes of resources required by the clients. For this, we provide the system architecture and relevant operations for the CDNs and evaluate the performance based on a simulation and compare it to the top existing techniques widely used in the current data centers around the globe.

1 Introduction

A cloud delivery network is like content delivery network, but in cloud delivery network, the server-level requests can be processed, and various applications can be deployed as well with seamless execution, whereas in content delivery network, the content is some form of audio, video, text, etc. A cloud delivery network can perform the same tasks as content delivery network with the above-mentioned add-ons. This makes the cloud delivery network an important topic for research in modern world (Zaidi, 2018). The resources or virtual machines.

A cloud delivery network is where various groups of servers are distributed in nature across the different parts of globe. The way a content delivery network works is that the data is replicated from on server in a data centre closer to the requested user base data centre to ensure better delivering of the resources to the end user. Cloud delivery network is a paradigm which works on the edge computing terminology where all the data centres are distributed globally to host different virtualised resources to the end users. Also, the architecture supports affordability and flexibility, hence making it one the most recent and important aspect to deliver resources to the end users who are distributed in nature. Cloud delivery networks are used to deliver virtualised resources to the end user economically and efficiently keeping the SLA intact (Ezugwu *et al.*, 2013). The resources can be in any form, such as images, sounds, audio, video, etc. The cloud delivery network leverages the virtualisation technology to provide with these resources to the end users. The main issue that any cloud delivery network owner would face is the cost of resource allocation and scalability (Datta and Venkata, 2015).

The below image shows the traditional cloud based content delivery network (Um *et al.*, 2014). From the image, there are two content providers controlling the control plane which consists of the data or resources requested by the end user. The cloud broker is the central form of communication between the control plane and the data plane. The data plane consists of various number of data centres distributed globally, which are managed by the cloud providers. The data plane communicates with the cloud broker and tells the available resources as per the end user requests. Then the cloud broker, forwards the message to the content provider and further the content is delivered to the end users (Thesis, Science and Fonville, 2014). The picture shows the flow of all the information with the help of three different marker lines.

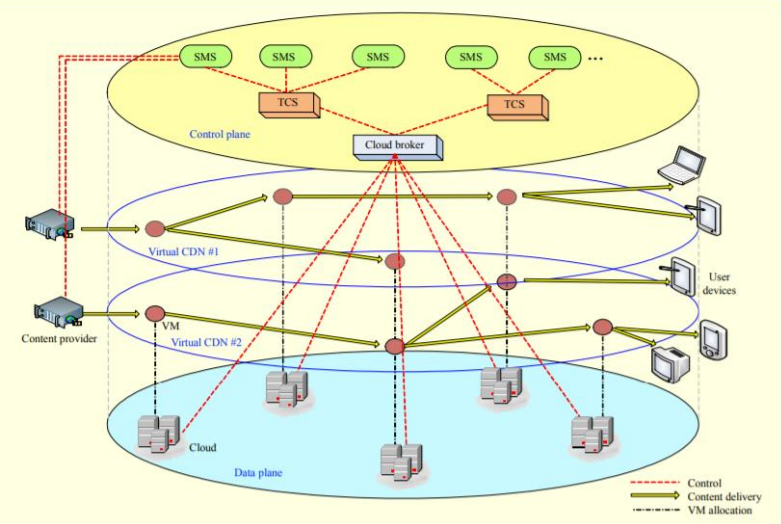


Figure 1 Cloud based CDN architecture (Um et al., 2014)

Cloud delivery network is a rapid enlarging domain in computer industry and research in modern times. The increasing demand in dynamic resources by end users have forced the CDN providers to provide the resources in a more elastic and simple manner based on the usage. The main objective of any content provider is to establish an efficient and effective load balancing technique which in turn helps in providing the resources to the end users expeditiously. As mentioned, load balancing is one of the main issue in cloud delivery networks as per the researchers (Filimban, 2017).

Typical cloud-base content delivery has components like an origin server, a request redirecting mechanism and a large number of cache servers known as Point of Presence (POP) (Wang *et al.*, 2015). The functionality of each of the above terms is explained below -

- Origin server: it is a storage system i.e. database which has all the content and data. When an end user requests for some specific resources, the main task of this origin server is to push the requested resources to POP which are spread across the global in nature. This origin server increases as well as maintains the scalability of the entire CCDN architecture.

- Request redirecting mechanism: the main function of the CCDN is to dynamically redirect the requested resources to the optimal edge cache server (node), closest to the user base where the request has been made by the end user, based on different and important parameters like response time, optimal path, bandwidth and latency. To achieve this dynamic redirection of resources for end users based on their

geographical region, there are number of methods available and which are the best available methods right now in the modern market. The following table describes the method by which the resources can be redirected –

- POP servers: POP servers are also known as the edge servers which are geographically located over different parts of the world. The main role of this POP servers is to provide with the content requested by the users around the globe. If a user requests the content and the POP server successfully provides with the requested data, and some other user requests the same data in around the same region for which this specific POP server is responsible to provide with the content, the POP server fetches the requested data from the origin server and stores in the repository for the user. But this doesn't happen all at the same time, sometimes, the requested data or resources are not available at the specific time. So, the POP server fetches the data and stores in the repository for a certain predefined time and the requesting user is redirected to the nearest POP server with where the resources requested are available. Also, the POP servers keep a track record of the users' activities over the time and keeps the resources in the repository accordingly. In this way the redirection of users can be avoided every time, thus increasing the flexibility, scalability and keeping the SLA intact.

Considering the above information regarding the introduction and the facts, there is a lot of scope in increasing the Quality of Service by decreasing the response time between the data centres and the user bases. Hence, the research question arises based on the theory –

What is the impact of Dynamic Load Balancer technique on Optimal Response Time for allocating virtual machines in a Cloud Delivery Network?

The main motivation behind this research topic is to reduce the overall response time between the user base and the data-centres with the increasing dynamic demand of the end users. We propose a new optimisation technique simulated in cloud analyst by running number of simulations for specific scenarios.

This report is organized as follows: In Section 2, literature review and discussion are presented. In Section 3, we present method and details on how our research is carried out. Section 4 explains the specification of our dynamic algorithm that we are proposing. In Section 5, we present the output results by performing different simulations on cloud analyst toolkit. In Section 6, we evaluate the numerical results obtained by comparing it with other techniques discussed in the section 2 of the paper and we summarize and conclude the paper with the future work in section 7.

2 Related Work

In this section, we discuss the existing work done on virtual machine/resource allocation in cloud delivery network which includes the topic of cloud computing and content delivery network. The cloud computing term is very wide which forms the basis for the issues of allocation in cloud delivery network (Samal and Mishra, 2013). The optimised allocation of

resources due to the increasingly dynamic changes of end users is the base study to improve the allocation, which is focused in this literature review.

2.1 Virtual machine placement in Cloud using variants of Round-Robin Algorithm

In a research paper (Samal and Mishra, 2013) the analysis of variants in Round Robin algorithms for virtual machine placement using load balancing in cloud computing. In this paper, the authors have performed experiments and have studied the outputs for the same for three different kinds of round robin algorithms. The three variants of round robin algorithms used in this paper are Round Robin (RR) and Modified Round Robin (MRR). The authors have also well-defined the advantages and disadvantages of these algorithm over each other. In a study performed by the authors, they propose a resource allocation technique with preemptable task execution. The disadvantage of this method is that it does not relate to cost optimization and time optimization. The only advantage of this method is it increases the optimization of utilization of clouds. So, the authors did another study, where they came up with MRR which had tweaking in their algorithm which overcame the disadvantages of the traditional RR algorithm. The changes in this method was, it was less pre-emptive, context switching which helped to reduce the overhead issues. In turn, it saved a lot of memory and space, which helped to decrease the response time in virtual machine allocation in cloud computing. The authors also suggested a criterion for task/resource scheduling, which is as follows:

- Context switching
- Throughput
- CPU utilization
- Turnaround time
- Waiting time
- Response time

The authors also performed experiments based on two of the three RR algorithms. The experiments were performed on the traditional RR algorithm and MRR algorithm. The MRR came out to be more efficient than the traditional RR algorithm.

Another paper (Mishra, 2014) suggests a priority based Round-Robin service broker algorithm for virtual machine allocation in cloud delivery network using cloud analyst as a simulator. In this paper the authors have performed various simulations in the cloud analyst using the round robin broker service algorithm. The authors have marked the issues with the round robin service broker algorithm, which includes the increase in response time due to the routing of the user requests because of the nature of the algorithm.

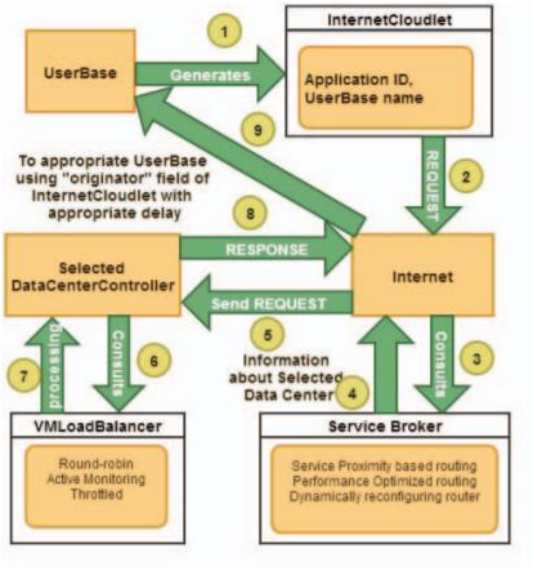


Figure 2 Routing of user requests problem in RR (Mishra, 2014)

From the figure above, the user requests are hopped to too many components before it is granted. This is the issue with the typical round robin algorithm as per the authors. This increases in the response time to allocate the desired resources as per the request. This decreases the quality of service affecting the Service Level Agreement (Utami and Fatta, 2017). To overcome this issue, the authors suggested a new proposal of Priority based Round-Robin selection algorithm which is based on the service proximity-based routing.
The proposed method works in two stages – pre-processing step and priority bases selection. The authors found out many issues in the service proximity-based routing because the data-centres were selected randomly which would affect the working and functioning of the method. So, based on this theory, they came up with the priority-based selection of data-centres. They defined the selection of data-centres which have the least processing time, response time and cost. The architecture of the proposed method is given below for better understanding.

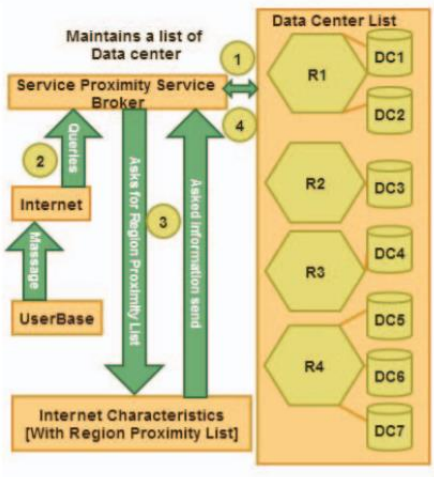


Figure 3 Service based proximity routing (Mishra, 2014)

The authors performed experiments and simulations in cloud analyst and compared the results of both the methods. The output in terms of overall response time and data-centre processing time were better for priority-based selection than that of the random selection method. But in terms of cost to the service provider and the end user, the priority method was costly than the random selection method.
In an academic thesis by (Filimban, 2017) the author has compared all the load balancing algorithms in cloud analyst. One of the algorithms in cloud analyst compared by the author is the round robin algorithm. The author has also defined the working of the Round Robin algorithm which is as follows (Filimban, 2017) –

1. Cloudlets of same size are created.
2. Cloudlet Coordinator devices the assigned Cloud task into same size of cloudlets.
3. Broker is created, and the task is assigned to the cloudlet co-ordinator.
4. Co-ordinator sends cloudlets to VM manager, in return the VM manager sends the list of required VMs.
5. Request for the exception of the cloudlet is set to the VM from the host.
6. FCFS scheduling policy is used for cloudlet scheduling in VM.
7. Sends the edited job as cloudlets in a wrapped file to VM mangers.
8. VM further passes the executed cloudlet as wrapped file to the cloudlet coordinator.
9. Cloudlet coordinator combines all excited cloudlets in wrapped file from combine to form the whole task.
10. Cloudlet coordinator sends the excited task in authenticated file format to the user client.
11. Results are printed.

The author also performed simulations based on four different scenarios where the data-centre placement and user base placement were different in cloud analyst. The author compared all the outputs and results with the other two algorithms from cloud analyst namely the throttled algorithm and the active VM placement algorithm.

2.2 Virtual machine placement in Cloud Delivery Network

In an academic paper by (Mills, Filliben and Dabrowski, 2011) the authors of compared in total of 18 VM-placement algorithms. The authors initially have shed some light on competing algorithms which are similar in nature. The authors have also taken the outputs after performing various experiments on these algorithms. Out of the 18 algorithms, they have shed some knowledge on the two-level VM placement algorithms. The authors have proved that the two-level VM placement algorithms form large clusters and these large clusters have a better response time than that of individual VM placement methods. They have also taken 42 total readings of all the experiments performed on these 18 algorithms and they have concluded that all the different algorithms yield small qualitative and quantitative changes in cloud content delivery network rand on-demand cloud, specially IaaS.

In academic thesis by (Filimban, 2017) the author thoroughly studied and compared the three algorithms pre-existing in the cloud analyst. The RR algorithm study is mentioned in the sub-section above. The other two VM placement algorithms that thee author compared are the throttled algorithm and the active VM monitoring algorithm. The author used these algorithms as they are widely used in the industry and the cloud analyst tool has them pre-loaded. The author ran simulations based on the four scenarios mentioned in the above sub-section. The author compared the output of all the three algorithms and found out the throttled algorithm had the best optimised response time when it comes to VM allocation

algorithms. The author did not suggest any new method or technique as the thesis was only based on comparative analysis of the VM allocation algorithms in the cloud analyst.

A dynamic resource scheduling method for virtual CDN was suggested in a paper by (Um *et al.*, 2014). In this paper, to overcome the VM allocation issue in CCDN or virtual CDN the authors proposed a new method which had a heuristic algorithm. The proposed method uses three classes of services namely Gold, Silver and Bronze. The authors used these three classes of services to produce a large overhead management for the resources requested by number of users. The algorithm used by the authors in this paper is mentioned below. The authors have well explained each step and the functioning of algorithm. The biggest advantage of this algorithm is that from time to time, the virtual CDN node examines the variation of arrival rate of the requests from the end users. The authors verified the rate of arrival and they set some pre-defined conditions and respectively they named their classes as gold, silver or bronze. As the content requests are classified according to their ranks of gold, silver and bronze, they are set in descending order in the queue. With this solution, the gold rank gets the highest priority and so on. This helps in execution in a priority manner, helping in reducing the overall response time between the virtual CDN and the end users.

2.3 VM allocation by heuristic algorithms

In an international journal by (Mohapatra and Rekha, 2013) the authors have evaluated and performed experiments in four heuristic algorithms namely, Round Robin (RR), Throttled, Equally spread current execution load (ESCEL) and First come first serve (FCFS). The authors have defined the functionality of all the four heuristic algorithms which are as follows -

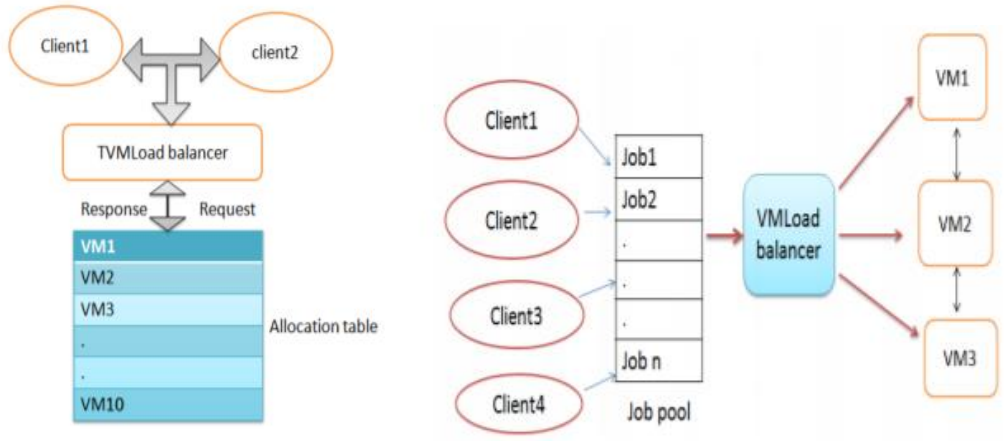


Figure 4 Scheduling of Throttled algorithm and ESCEL algorithm

After evaluating the outputs from all the four heuristic algorithms, the authors came to a conclusion that the Round-Robin algorithm has the best optimal time out of all the four heuristic algorithms.

In another paper by (Reddy *et al.*, 2016) the authors have performed analysis on different heuristic load balancing algorithms. The authors have taken heuristic algorithms like FCFS, Genetic algorithm (GA), Multi-objective model for scheduling and Ant colony optimization (ACO). The authors have selected these three heuristic algorithms as they are

well known to decrease the load balancing problem in cloud computing environment. The authors have performed experiments on the above algorithms and they came to a conclusion that the load balancing problem can be further reduced, hence they proposed a new heuristic algorithm called Multi Objective Ant Colony Algorithm (MOACA). They simulated the experiments based on the new MOACA algorithms and they found out to reduce the problem of load balancing in cloud computing environemt by further reducing the total response time.

2.4 Discussion

The thorough research in the work done for VM placement issues gives us better data to consider the important elements and methods to improve this QoS attribute. With the help of above literature review on VM placement issues in cloud computing, we have formulated a discussion table which is an overview of the literature review.

Table 1 Discussion of Literature Review

Literature Review Tittle	Problems addressed
Analysis of variants in Round Robin Algorithms for load balancing in Cloud Computing	Less Scalability
Priority Based Round-Robin Service Broker Algorithm for Cloud-Analyst	High response time
Comparing VM-Placement Algorithms for On-Demand Clouds	High data-centre processing time
An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing environment	High cost
Comparison Study of The Most Common Virtual Machine Load Balancing Algorithms in Large-Scale Cloud Environment Using Cloud Simulator	No comparison to other load balancing algorithms
Dynamic Resource Allocation and Scheduling for Cloud-Based Virtual Content Delivery Networks	High cost
Comparative Analysis of Load Balancing Algorithms in Cloud Computing	high Latency
A Comparison of Four Popular Heuristics for Load Balancing of Virtual Machines in Cloud Computing	High response time
Load Balancing in Cloud Computing Using Dynamic Load Management Algorithm	High bandwidth
Performance Analysis of Load Balancing Algorithms in Cloud Computing Environment	High throughput

After evaluating the problems addressed in the above literature review table, there should be some defined attributes which have to be considered to solve the VM placement problem in Cloud Delivery Network. The following parameters can be considered after evaluating the literature review discussion table –

- High scalability
- Less Overall response time
- High data-centre processing speeds
- Low cost

The above parameters can be considered to build a new method which can help to tackle the VM placement issue in Cloud Computing. In the next section, we will propose our method which can help to reduce the overall response time keeping the data-centre processing speed high at the same cost or maybe cheaper than previous methods.

3 Research Methodology

To improve the problem of virtual machine allocation in cloud computing environment i.e. the Cloud Delivery Network, we propose a new technique called "Dynamic Load Balancer (DLB)". To prove the efficiency of our algorithm, we will compare the simulations with the two market top techniques namely the Round-Robin (RR) and the Equally Spread Current Execution Load (ESCEL) (Verma *et al.*, 2016)

In this section, the process flow of the proposed technique and its working has been explained. As mentioned above in section 2, we are proposing a new algorithm called "Dynamic Load Balancer", which can help in reduce the optimum time taken to allocate the resources/virtual machines to the end user. This will solve the problem of load balancing in cloud delivery network.

The below diagram shows the process method or flow of the proposed technique. The technique has three main phases –

In phase 1, the technique recognises and predicts the dynamic requests from the end users. After processing the dynamic requests from the end user, state of the virtual machines is checked. If any virtual machine is not fit for allocation based on the pre-defined parameters, it is discarded at that very instance. The rest of the virtual machines which are fit for allocation are given Virtual Machine ID which is unique.

In phase 2, the virtual machines are allocated to the end users based on their request. The technique also checks, whether the request have been completely fulfilled or the request is overloaded or not. If the resources are under-allocated, then additional virtual machines are allocated to the same request and if there are extra virtual machines, they are discarded.

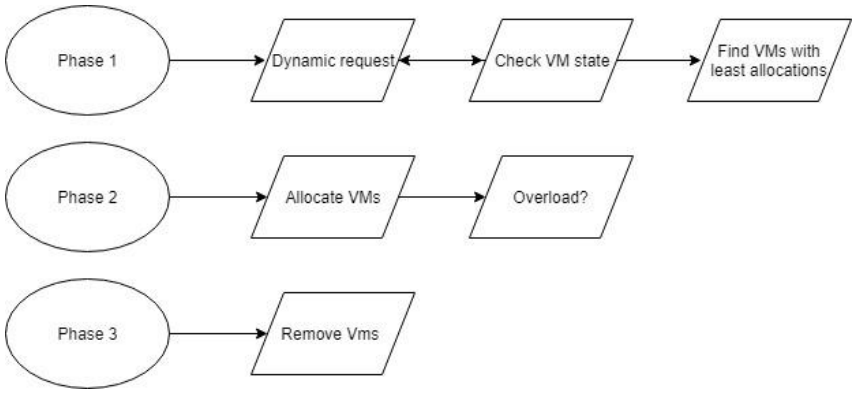


Figure 5 Process method

In phase 3, the output is recorded of the total time taken to allocate the requested resources from the data centre to the user base. The optimal response time can be recorded and be compared to the top traditional techniques currently used by the industries.

3.1 Flow chart stages

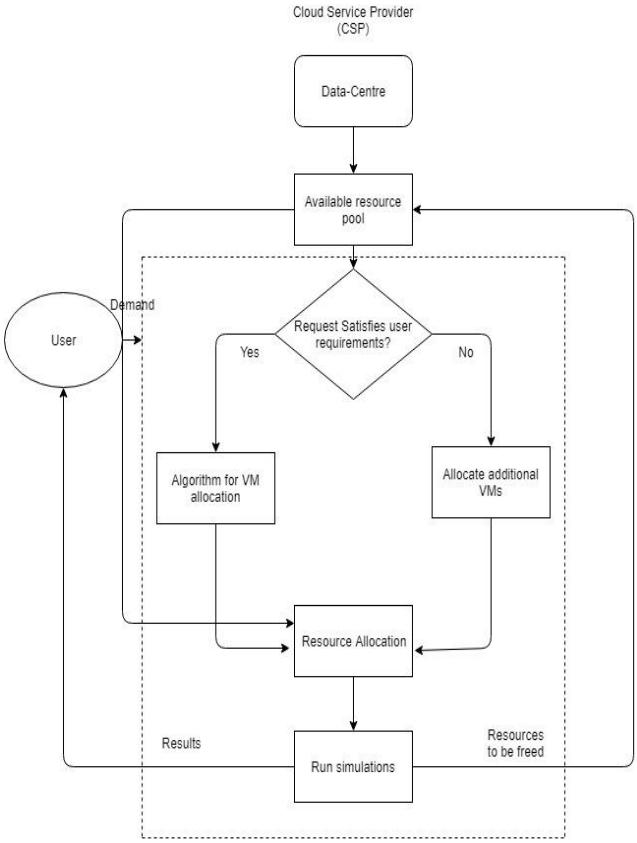


Figure 6 Flow Chart of the process

In the figure above, the three main components are the cloud service provider (CSP), the user and the simulator i.e. Cloud Analyst. There is a resource pool available with the CSP, where all the information regarding the resources available are present. There is a dynamic pool of resources available with the CSP to cater the dynamic changing needs of the users.

3.2 Stage 2 – Sequence diagram

In this section, the sequence diagram for the entire working architecture has been explained. The main components of the sequence diagram are the user, host, VM, application and the simulator i.e. Cloud Analyst in this case.

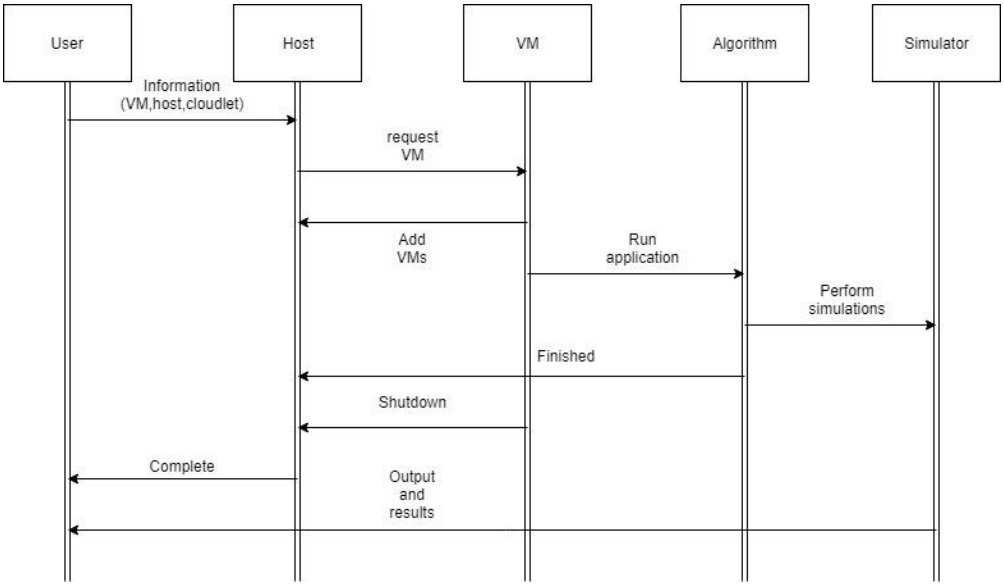


Figure 7 Sequence Diagram of the proposed technique

Step 1: user requests number of virtual machines/resources based on the requirement
Step 2: the host allocated the resources accordingly from the parent data center
Step 3: if there are a smaller number of VMs allocated, the remaining VMs are provided to the host
Step 4: the application is run based on the algorithm in the simulator
Step 5: number of simulations are performed to obtain the desired output and results
Step 6: simulations are run, and the application notifies the host
Step 7: VMs are shut down after the completion of the simulations
Step 8: the host notifies the user about the simulation
Step 9: the outputs are displayed directly to the user on the simulator

3.3 Method Complexity

The response time optimization factor of this method is determined by a linear equation. The complexity is also because of the resources available in the data-centre. (Laila and Kumar, 2018) suggested a similar linear equation which can help in determining the response time optimization factor.
Formula –
Response time optimization = total allocated resources - total missing resources divided by total allocated resources*100

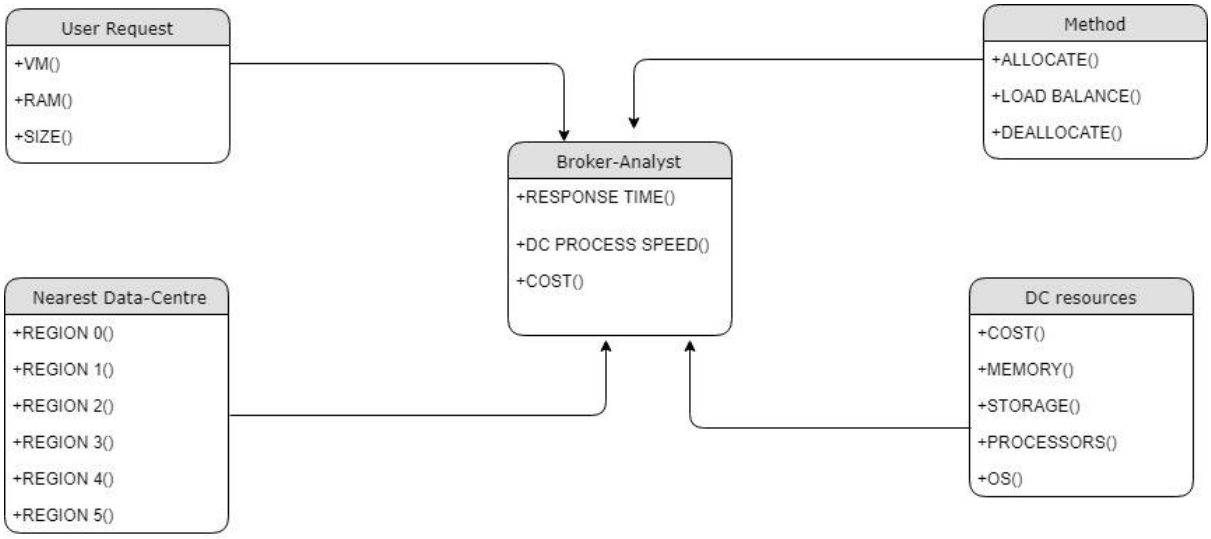


Figure 8 Object-Class Diagram

The above figure shows the object-class diagram for the proposed method. There are total five classes and they have their individual objects. The broker-analyst is the main object which overlooks the entire functioning of the method. The other four object gives the data and information to the broker-analyst.
The complexity of the method comes when the end user requests certain resources from the cloud delivery network provider and those resources are not available in the nearest data-center. During such scenarios, the end user is re-directed to the other available data-center without checking for the required resources (Seyfabad, 2015). The CSP cannot add resources at that very moment, hence making it high time-consuming procedure. And in the other case, when the end user is re-directed to another nearest data-center, the latency bandwidth issues comes into play.

4 Design Specification

In this section, the proposed architecture which consists of Cloud Analyst and the working along with the architecture has been explained (Acharya, 2017).

4.1 Prototype architecture

We have created a prototype architecture which is integrated with the Eclipse IDE.

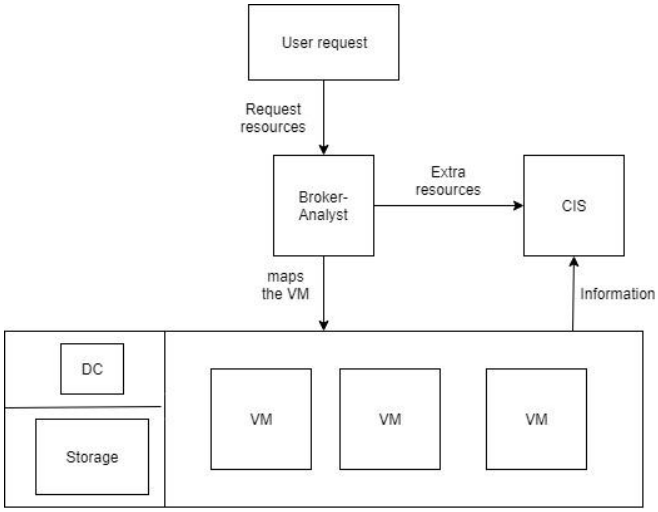


Figure 9 Prototype architecture

The above figure is the proposed architecture of cloud analyst for the research project. In this proposed architecture, the broker-analyst is one of the most important entities. The user requests the number of resources as desired from the broker-analyst. There can be a few data-centres available with the provider. In the above proposed architecture, the data-centre defined is closest to the user base. The data-centre has number of individual attributes inside it like storage, memory, number of hardware devices, etc. The broker-analyst maps the requested virtual machine on the data-centre. All the DC information is present with the cloud information service (CIS). If the mapping of the required resources are less than the user requested resources, the broker-analyst goes to the CIS and checks for the information and if needed requests the extra resources as well. When all the conditions are fulfilled of allocating the resources, the user gets the resources.

5 Implementation

In this section, we discuss out implementation part of the proposed method described in the section 3. The tools and their important attributes who contribute to perform the simulations have been well defined in this section.

5.1 Cloud Analyst toolkit

Cloud Analyst is a tool which is based on Cloud Sim architecture. It is a GUI tool. It was developed at the University of Melbourne. The main aim of this tool is to perform number of simulations for virtual machine allocation using load balancing algorithms. With the internet modelling and behavior of internet applications, the cloud analyst tool is created on topmost of cloud sim. (Pan, 2015). Also, the data-center and user base configurations needs to be done along with the broker configuration before executing the simulations.
The basic and important components of the Cloud Analyst tool are as follows (Pan, 2015)-

- GUI Packages
- User Base
- Internet
- DataCentreController

- VmLoadBalancer
- Simulation
- CloudAppServiceBroker

5.2 Eclipse IDE

Eclipse is an IDE which has a base workspace and an extensible plug-in system for customizing the environment based on the dynamic changes. It is the most widely used Java IDE in computer programming (Wang et al., 2015).

5.4 Java 1.7

Java 1.7 version is required to install for cloud analyst to work properly. This version enables us to code in integrated developer platform(eclipse) and create our own technique which can be simulated on cloud analyst platform. This java version is open source and can be downloaded from java website.

5.5 Front end GUI of Cloud Analyst

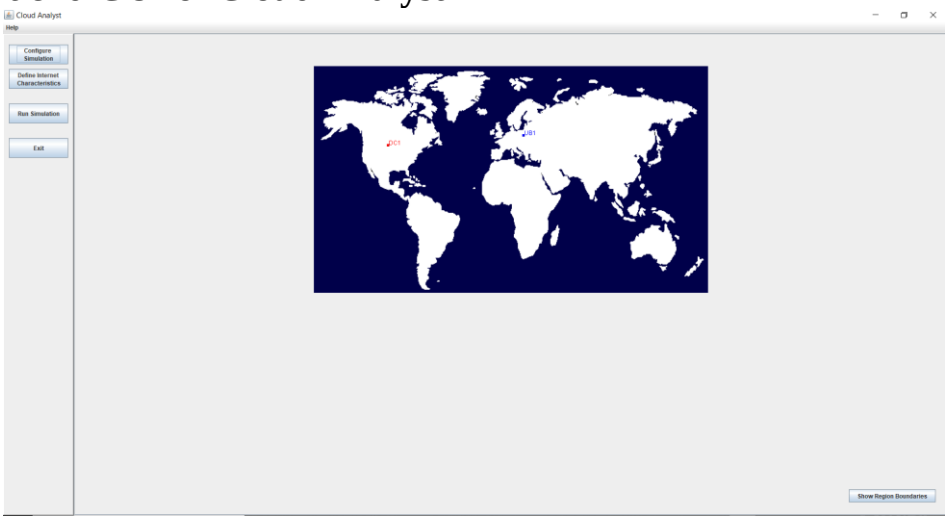


Figure 10 Front end GUI of Cloud Analyst (Mohapatra and Rekha, 2013)

The above figure is the front GUI of Cloud Analyst which can be accessed by both i.e. the CSP and the end users. There are various functions which are available in the cloud analyst to run various kinds of simulations with different parameters. The data centers and the user bases can be spread across five different geographical regions in cloud analyst. The data center configuration along with the user base and broker configuration must be done as per the requirement before running the simulations. The output from the simulations are obtained in the form of graphs and tables which is in detail and a GUI output which is the overview and short result of the simulations run can be displayed (Mohapatra and Rekha, 2013).

6 Evaluation

In this section, we present and discuss the experiment scenarios we have performed to get the desired output which solves the problem of the optimal response time between data-center and user base in the virtual machine allocation problem with the help of load

balancing. The setup of user base and data-centers can be pictured as follows in the Cloud Analyst (Nitika, 2016). The individual configurations can be changed as per the requirement of the end user and the data-centers and user bases can be allocated in any of the six different geographically distributed regions.

6.1 Scenario 1

In scenario 1, there are two data-centers which are distributed in corner regions of the world. Along with two data-centers, we have established 8 user bases which are geographically distributed across the globe in six geographical regions. The experiment run in this scenario is to optimize the response time between the data-centers and the user bases. The scenario is run for a total of 60 hours and there are in total four iterations i.e. the round-robin algorithm, the equally spread current execution load algorithm and the Dynamic Load Balancer (DLB). The detailed results have been captured and interpreted as follows –

Iteration 1 – Round-Robin Algorithm (RR) –

Response Time by Region –

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.14	35.43	66.17
UB2	200.20	140.39	264.26
UB3	200.12	147.22	260.14
UB4	300.24	211.59	545.13
UB5	300.18	213.12	482.63
UB6	300.11	216.12	550.13
UB7	300.20	208.62	577.63
UB8	200.18	143.14	448.10

From the above table, the average, minimum and maximum response time can be seen between each of the user base and the data-centers. The minimum time means the data-center is closest to that user base and if the time is greater than that of minimum, that means user base is far away from the data-centers. Response time is measured in milliseconds (ms).

Overall response time -

	Avg (ms)	Min (ms)	Max (ms)
Overall response time	232.51	35.43	577.63
Data-centre processing time	0.31	0.01	1.01

The data-center processing time is how much requests can a data-center process. The average, minimum and maximum data-center processing time can be seen from the above table. Response time is measured in milliseconds (ms).

Iteration 2 - Equally Spread Current Execution Load algorithm (ESCEL)–

Response Time by Region –

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.14	34.43	66.17
UB2	199.22	139.39	264.26
UB3	200.13	147.22	260.14
UB4	301.23	212.59	545.13
UB5	302.20	215.12	482.63
UB6	300.09	216.12	550.13
UB7	298.20	206.62	575.63
UB8	200.18	143.14	448.10

Overall response time –

	Avg (ms)	Min (ms)	Max (ms)
Overall response time	235.51	34.43	575.63
Data-centre processing time	0.31	0.01	1.00

Iteration 3- Dynamic Load Balancer (DLB) –

Response time by region –

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	49.94	33.43	66.17
UB2	200.21	144.27	264.26
UB3	200.11	140.39	260.14
UB4	300.26	211.59	545.13
UB5	298.20	211.12	482.63
UB6	300.11	215.12	551.13
UB7	300.20	205.62	574.63
UB8	200.18	143.14	448.10

Overall response time –

	Avg (ms)	Min (ms)	Max (ms)
Overall response time	229.51	33.43	574.63
Data-centre processing time	0.31	0.01	1.00

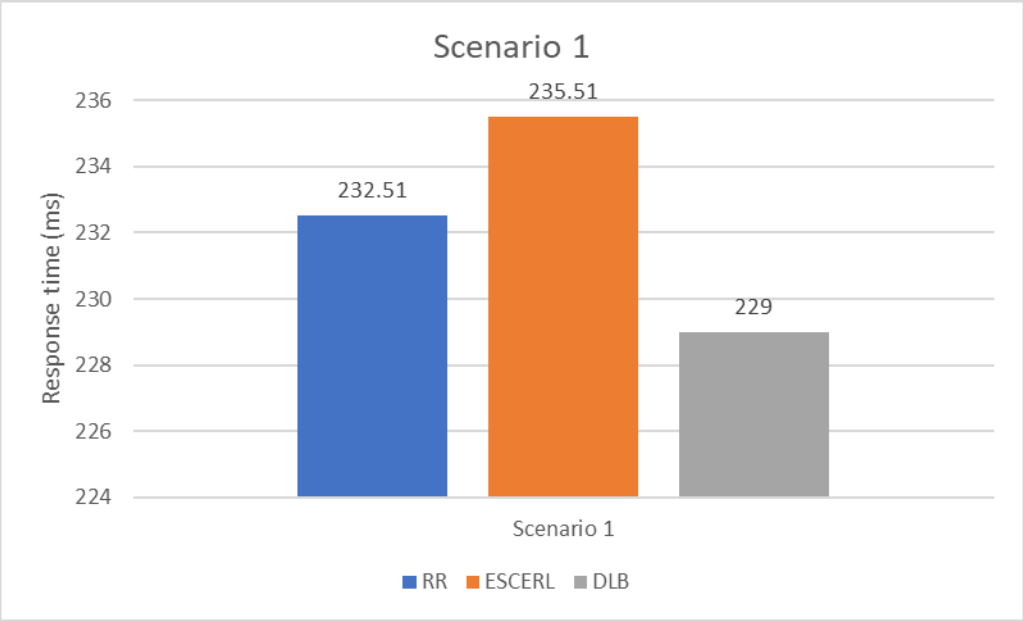


Figure 11 Graph for response time of scenario 1

6.2 Scenario 2

In scenario 2, the data-centers are placed in the middle of the distributed region, so that the user base can request resources from the two data-centers and the overall response time can be recorded. All the configurations are kept same as described in scenario 1. The response time is measure in milliseconds (ms). In this scenario we have tabulated the results of all three iterations in one table itself.

Response time by region –

Userbase	Iteration 1 (RR)			Iteration 2 (ESCEL)			Iteration 3 (DLB)		
	Avg (ms)	Min (ms)	Max (ms)	Avg (ms)	Min (ms)	Max (ms)	Avg (ms)	Min (ms)	Max (ms)
UB1	300.09	211.73	567.60	300.10	211.73	557.57	300.09	211.73	567.60
UB2	501.13	350.39	850.60	501.10	352.64	850.60	501.09	350.39	812.10
UB3	500.88	367.72	840.10	500.80	350.23	833.10	500.88	367.72	840.10
UB4	300.24	211.60	390.14	300.22	211.60	390.14	300.23	211.60	390.14
UB5	300.18	213.15	396.14	300.21	213.15	394.76	300.19	213.15	394.76
UB6	300.11	216.14	391.64	300.11	216.14	391.64	300.12	216.14	391.64
UB7	50.27	34.89	66.60	50.26	34.89	64.65	50.27	34.89	66.60
UB8	400.15	286.14	530.24	400.18	286.14	530.24	400.18	286.14	530.2

The above table gives all the response times for the three different iterations simulated to get the desired output results. Also, from the figure below, we can see how the detailed results are displayed on the front-end GUI screen for better and easy pictorial representation.

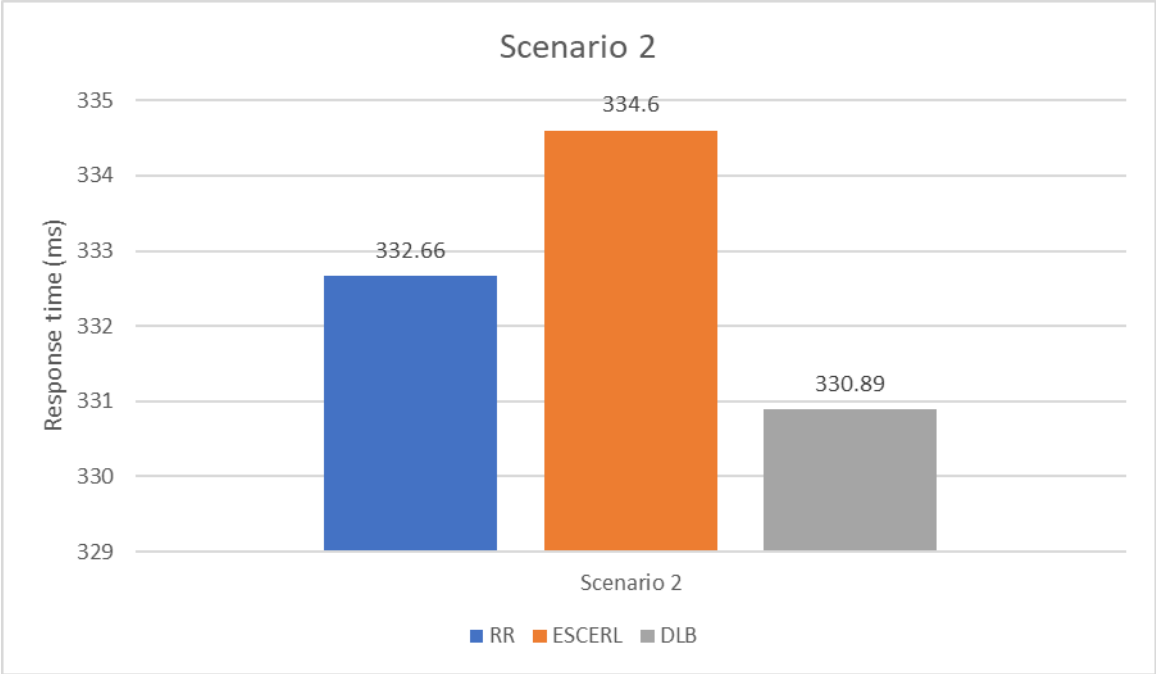


Figure 12 Graph for response time of scenario 2

6.3 Scenario 3

In this scenario, we have kept all the user base in the same region as the above two scenarios, but instead of two data-centers geographically distributed around the globe, we have taken six different data-centers with the same configuration from above two scenarios. We have distributed one data-center to each region. In this case as well, the response time is measured in milliseconds (ms).

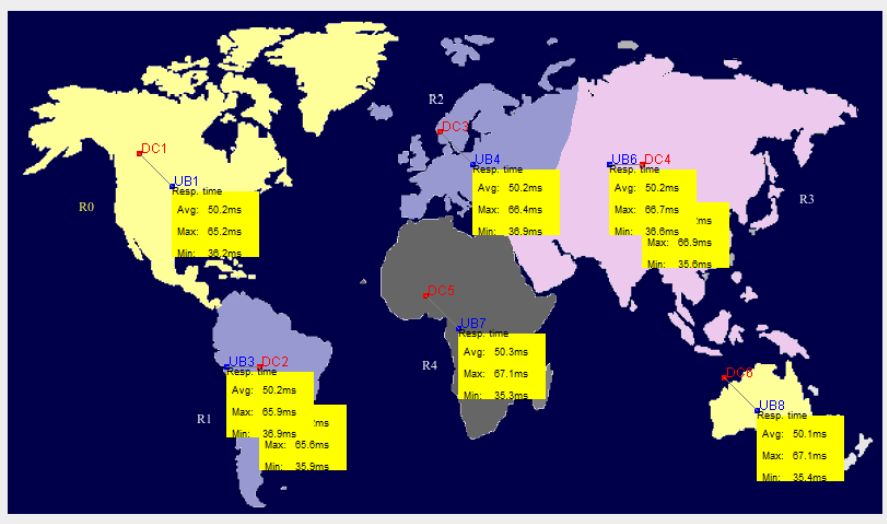
We have tabulated all the three iterations in one table for better picture presentation and understanding.

Userbase	Iteration 1 (RR)			Iteration 2 (ESCEL)			Iteration 3 (DLB)		
	Avg (ms)	Min (ms)	Max (ms)	Avg (ms)	Min (ms)	Max (ms)	Avg (ms)	Min (ms)	Max (ms)
UB1	54.15	36.18	65.18	51.16	36.18	65.18	49.55	35.20	64.55
UB2	54.22	35.89	65.61	52.22	35.89	65.61	49.20	35.55	64.63
UB3	55.22	36.89	65.90	52.22	36.89	65.90	49.60	35.66	64.20
UB4	55.18	36.92	66.38	51.17	36.92	66.38	49.88	35.23	64.02
UB5	53.17	35.64	66.88	51.17	35.64	66.88	48.98	34.98	63.95
UB6	53.16	36.62	66.72	51.16	36.62	66.72	47.55	34.21	62.21
UB7	54.28	35.32	67.10	52.27	35.32	67.10	48.65	35.68	64.89
UB8	55.14	35.38	67.13	51.14	35.38	67.13	49.50	36.00	63.50

The above table shows the response time for all the three iterations and it can be clearly seen that the time of iteration 3 (DLB) which is the proposed algorithm to improve the virtual machine allocation problem using load balancing in Cloud Delivery Network. The below table shows the overall response time of the iteration 3 (DLB).

	Avg (ms)	Min (ms)	Max (ms)
Overall response time	50.19	35.32	67.13
Data-center processing time	0.52	0.01	1.12

From the above table, compared to the previous two scenarios, the data-center processing time is also efficient because of the placement of the data-centers from the user base. The below image shows the graphical output which can used to interpret results in gist and can be used to formulate tables and generate different graphs based on the data.



So, from the above three scenarios where we simulated different iterations to see the effect of Dynamic Load Balancer (DLB) on virtual machine allocation in Cloud Delivery Network (CDN), we have formulated an overall graph which gives the overall detail in short which can be useful for pictorial representation and interpretation.
In the below graph, we have considered the average response time of all the scenarios and their iterations, as average time can determine the best iteration out of all the three. For better understanding, the graph is formulated along with the table and the values of all the average response time. The response time is measured in milliseconds (ms).

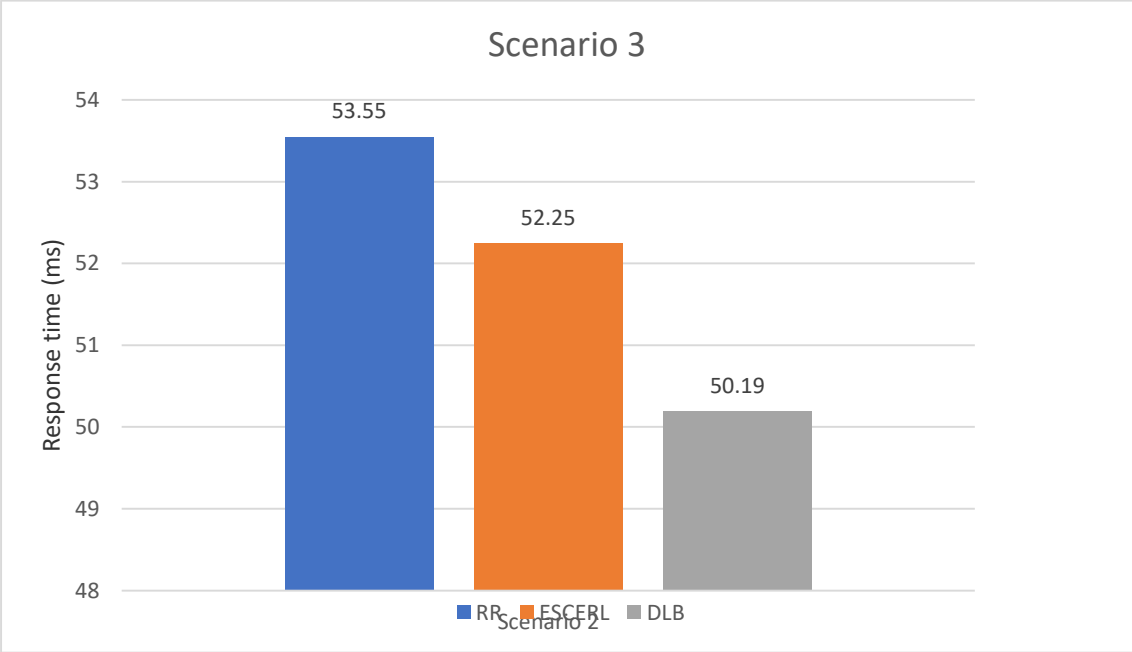


Figure 13 Graph for response time of scenario 3

6.4 Discussion

With the chosen method, we were able to successfully create a cloud delivery network environment in cloud analyst. The simulations were performed and compared to the two traditional methods of Round Robin and ESCEL methods. The output of our technique was promising in terms of reducing the overall response time between the user base and data-centres, which was the goal. For further realistic environments, this technique can be integrated with actual cloud platforms, where the outputs will be accurate and machine learning algorithms can be used instead of traditional algorithms.

7 Conclusion and Future Work

This research project is focusing on the effect of Dynamic Load Balancer algorithm compared to the existing heuristic algorithms on virtual machine placement in cloud delivery network. Two ideas clubbed together can be called a hybrid method or a meta-heuristic algorithm.

The proposed method not only focuses on reducing the response time, but also the data-center processing speeds which can directly reduce the overall response time between the data-centers and user bases. The cost factor in this proposed method is same as to the cost compared to the traditional heuristic methods. The future work in this research project can be addition of actual cloud platforms, where the simulations can run in real time and better results can be obtained. Also, machine learning algorithms can be used for dynamic load balancing as the user requests are dynamic in nature. Also, the cost can be reduced further by considering the cost parameters in the method.

8 Acknowledgement

I express my gratitude and thanks to Prof. Victor Del Rosal for meeting me every week and giving me guidance and feedback for my research project. He helped me in every stage of this project and guided me and pointed me specifics expected at this level of research project. I want to thank my family and friends who supported me and kept me motivated to complete this research project. At last, I want to thank National College of Ireland for providing me the platform and opportunity to pursue my goals.

References

Acharya, S. (2017) 'A COMPARATIVE STUDY ON RESOURCE ALLOCATION', (2), pp. 68–73.

Datta, S. and Venkata, V. (2015) 'Cost Aware Virtual Content Delivery Network for Streaming Multimedia', (October).

Ezugwu, A. E. *et al.* (2013) 'Virtual Machine Allocation in Cloud Computing Environment', 3(June), pp. 47–60. doi: 10.4018/ijcac.2013040105.

Filimban, R. (2017) 'Comparison Study of The Most Common Virtual Machine Load Balancing Algorithms in Large-Scale Cloud Environment Using Cloud Simulator Comparison Study of The Most Common Virtual Machine Load Balancing Algorithms in Large-Scale Cloud Environment Using Cloud Simulator'. doi: 10.21427/D78S44.

Laila, M. A. and Kumar, A. A. (2018) 'An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing', *Cluster Computing*. Springer US. doi: 10.1007/s10586-018-1769-z.

Mills, K., Filliben, J. and Dabrowski, C. (2011) 'Comparing VM-Placement Algorithms for On-Demand Clouds'. doi: 10.1109/CloudCom.2011.22.

Mishra, R. K. (2014) 'Priority Based Round-Robin Service Broker Algorithm For Cloud-Analyst', pp. 878–881.

Mohapatra, S. and Rekha, K. S. (2013) 'A Comparison of Four Popular Heuristics for Load Balancing of Virtual Machines in Cloud Computing', 68(6), pp. 33–38.

Nitika, M. (2014) 'Comparative Analysis of Load Balancing Algorithms in Cloud Computing', 1(1), pp. 34–38.

Pan, R. (2015) 'Load Balancing in Cloud Computing Using Dynamic Load Management Algorithm', pp. 773–778. doi: 10.1109/ICGCIoT.2015.7380567.

Reddy, V. K. *et al.* (2016) 'Performance Analysis of Load Balancing Algorithms in Cloud Computing Environment', (November). doi: 10.17485/ijst/2016/v9i18/90697.

Samal, P. and Mishra, P. (2013) 'Analysis of variants in Round Robin Algorithms for load

balancing in Cloud Computing', 4(3), pp. 416–419.

Seyfabad, M. S. (2015) 'Virtual Machine Allocation in P2P-Cloud Live Video Straming'. doi: 10.1109/3PGCIC.2015.67.

Thesis, M., Science, C. and Fonville, M. (2014) 'The Virtual Machine Delivery Network'. Um, T. *et al.* (2014) 'Dynamic Resource Allocation and Scheduling for Cloud-Based Virtual Content Delivery Networks', 36(2). doi: 10.4218/etrij.14.2113.0085.

Utami, E. and Fatta, H. A. (2017) 'Dynamic virtual machine allocation policy in cloud computing complying with service level agreement using CloudSim Dynamic virtual machine allocation policy in cloud computing complying with service level agreement using CloudSim'. doi: 10.1088/1757-899X/263/4/042016.

Verma, M. *et al.* (2016) 'Dynamic resource demand prediction and allocation in multi-tenant'. doi: 10.1002/cpe.

Wang, M. *et al.* (2015) 'An Overview of Cloud Based Content Delivery Networks : Research Dimensions An Overview of Cloud Based Content Delivery Networks : Research Dimensions and State-of-the-Art', (May 2017). doi: 10.1007/978-3-662-46703-9.

Zaidi, T. (2018) 'Virtual Machine Allocation Policy in Cloud Computing Environment using CloudSim', 8(1), pp. 344–354. doi: 10.11591/ijece.v8i1.pp344-354.

Appendix

Configuration Manual

MSc Research Project
Cloud Computing

Deep Shah
Student ID: X17141702

School of Computing
National College of Ireland

Supervisor: Victor Del Rosal

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	**Deep Shah**		
Student ID:	**X17141702**		
Programme:	**MSc. Cloud Computing**	**Year:**	**2019-2020**
Module:	**MSc. Research Project**		
Lecturer:	**Victor Del Rosal**		
Submission Due Date:	**18/04/2019**		
Project Title:	**Efficient virtual machine allocation and optimization of response time in cloud delivery network**		
Word Count:	**722**	**Page Count:**	**7**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:

Date: **18/04/2019**

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if	

applicable):	

Configuration Manual

Deep Shah
X17141702

1 Introduction

This configuration manual guides through all the steps and processes of implementation of the research project. It includes steps from the installation, configuration and implementation of the software and platform required to run the desired simulations in the Cloud Analyst tool. It is also a manual guide for the testbed setup and code for method.

2 Setting up the test environment

2.1 Hardware specification

CPU	Intel 7700HQ - 2.28Ghz
RAM	16 GB

2.2 Software specification

Operating System	Windows 10 Home
Cloud Analyst	1.0
Eclipse IDE	4.10.0
Java	1.7

2.3 Data set

We have used the pre-defined data set which is already present in Cloud Analyst by planet lab.

2.4 Installing the Software

2.4.1 Installing Eclipse IDE on Windows 10

- Download Eclipse IDE latest version from https://www.eclipse.org/downloads/
- Copy and extract to the C drive in the Windows 10.
- Follow the installation process and give the desired path of installation.

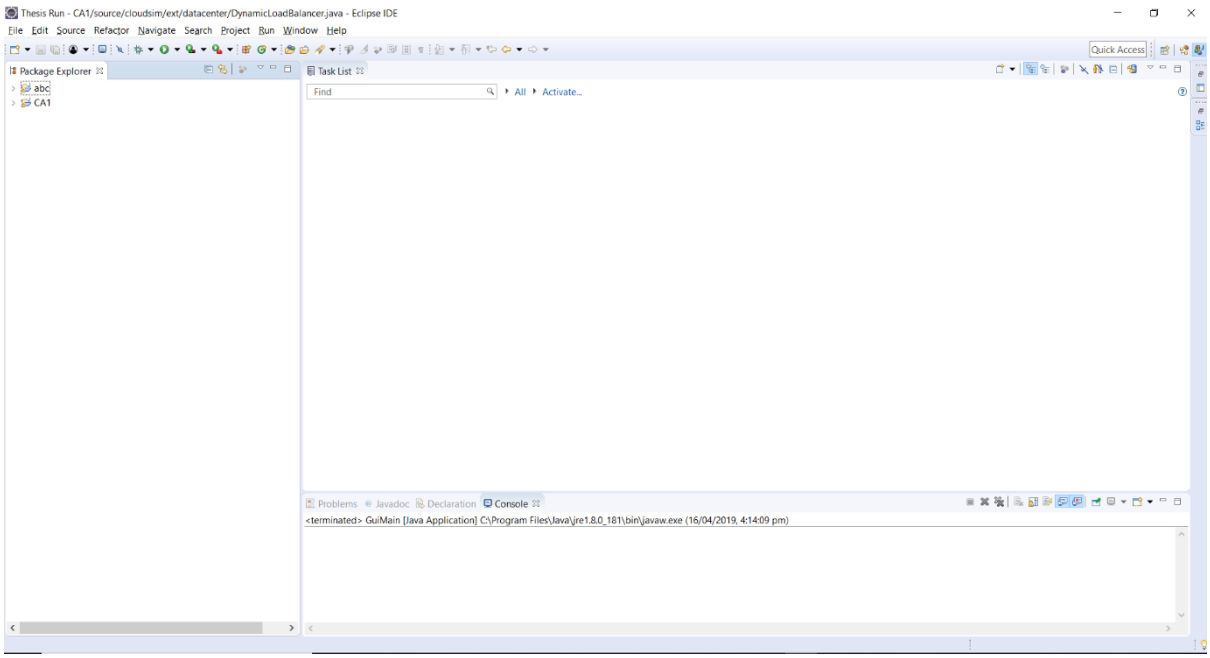


Figure 1 Eclipse IDE GUI

2.4.2 Installing Cloud Analyst in Eclipse IDE

- Download cloud analyst zip file from http://cloudsim-setup.blogspot.com/2013/01/running-and-using-cloud-analyst.html
- Extract the contents of the zip file in a folder on any drive on windows 10.
- Downlaod the common-math3-3.6.1 zip file from http://cloudsim-setup.blogspot.com/2013/01/running-and-using-cloud-analyst.html
- Extract the contents of the zip file in a new folder in any drive on windows 10.
- Install Java version 1.7 if not installed on the system
- Open Eclipse IDE
- Click on New → Java project

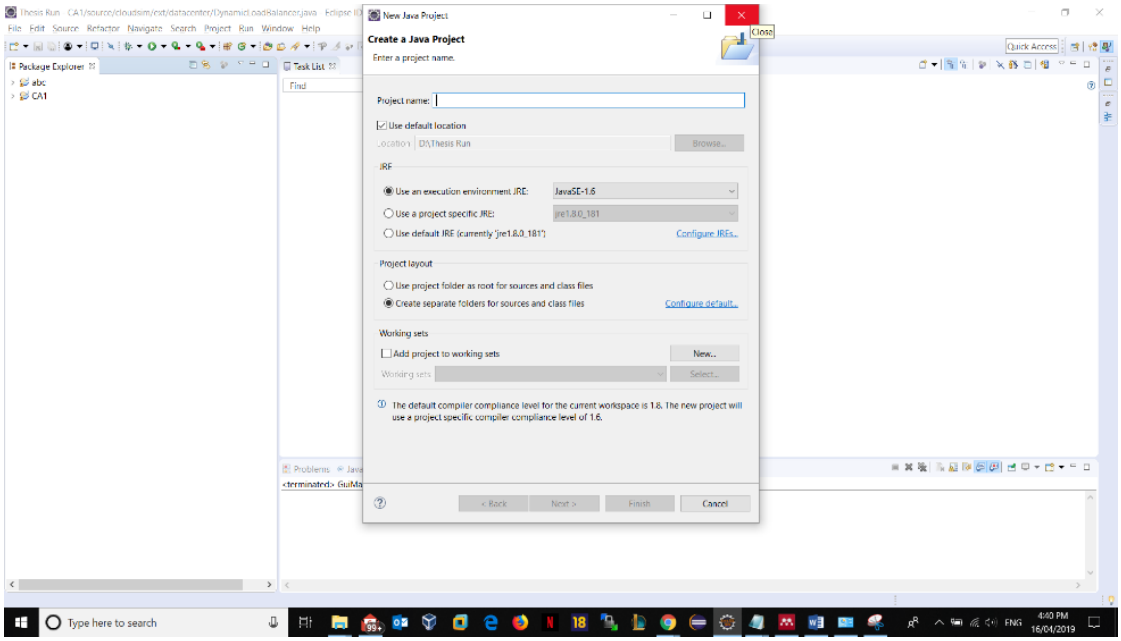


Figure 2 Create a new Java Project

- Give the name of your desired project name and click finish
- Then go to file → import

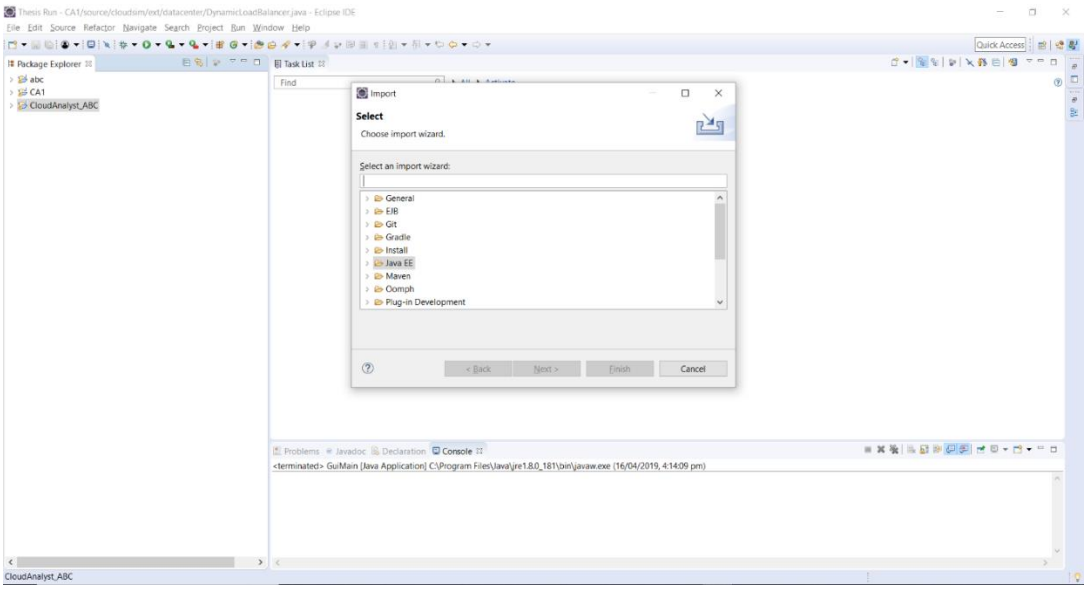


Figure 3 Import files into project

- Select the folder where you have extracted the zip file and click select all files.
- Click finish.
- Cloud Analyst is successfully installed in Eclipse IDE

2.5 Running Cloud Analyst v1.0 in Eclipse

- Go to the project files → cloudsim.ext.gui → GuiMain.java
- Right click the file name and run as → java project

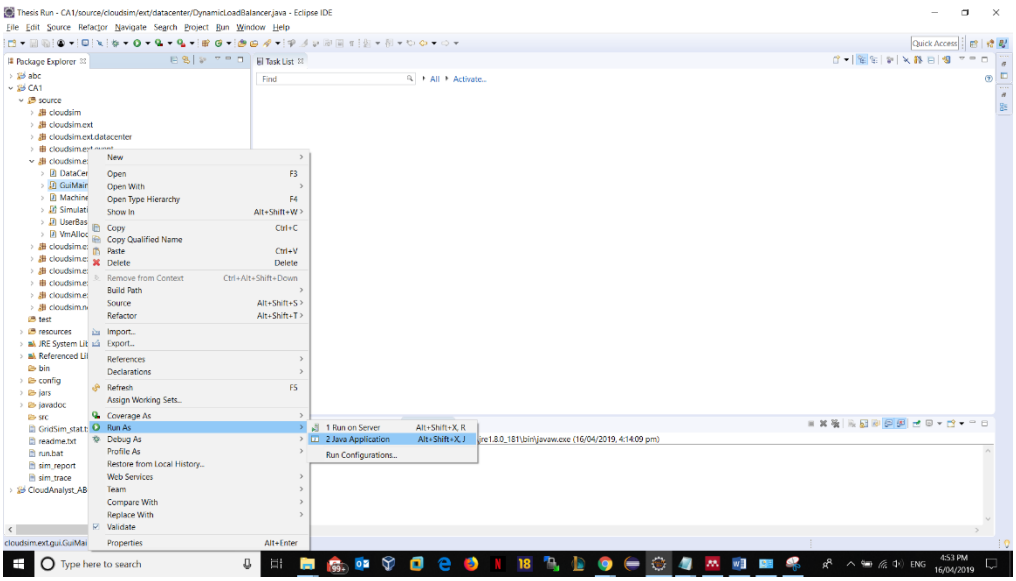


Figure 4 Run Cloud Analyst

- The GUI of cloud analyst will be visible

Figure 5 GUI of Cloud Analyst

2. 6 Configuring Cloud Sim and Designing new method

- Click on the configure simulation button

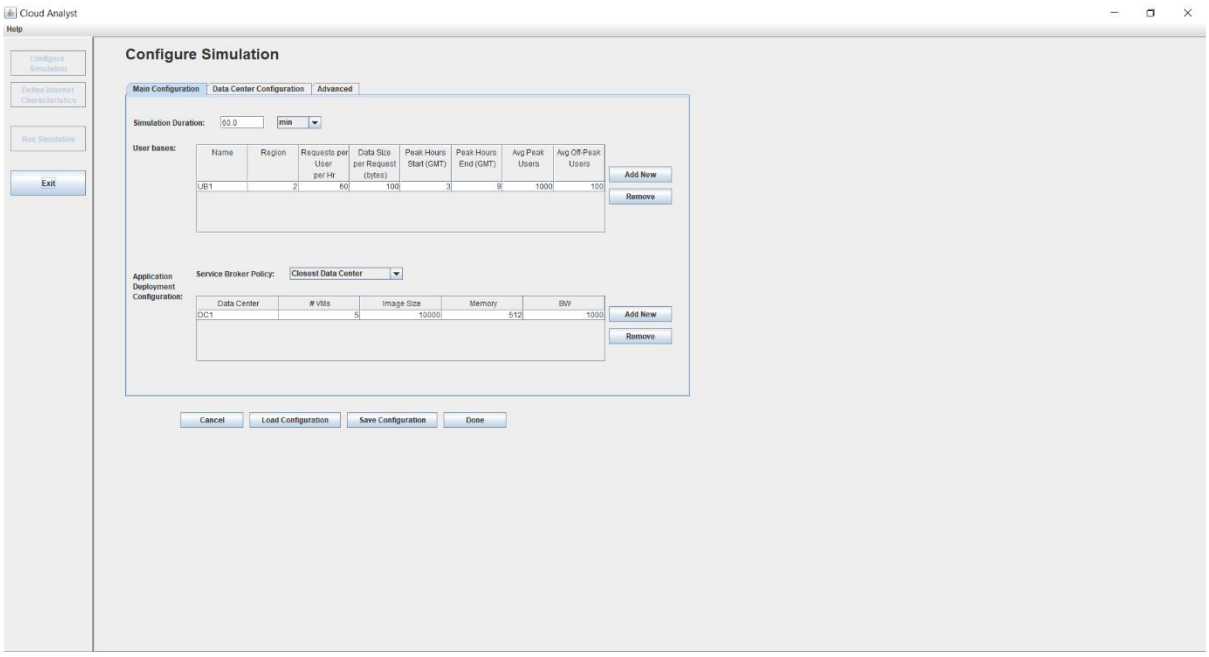
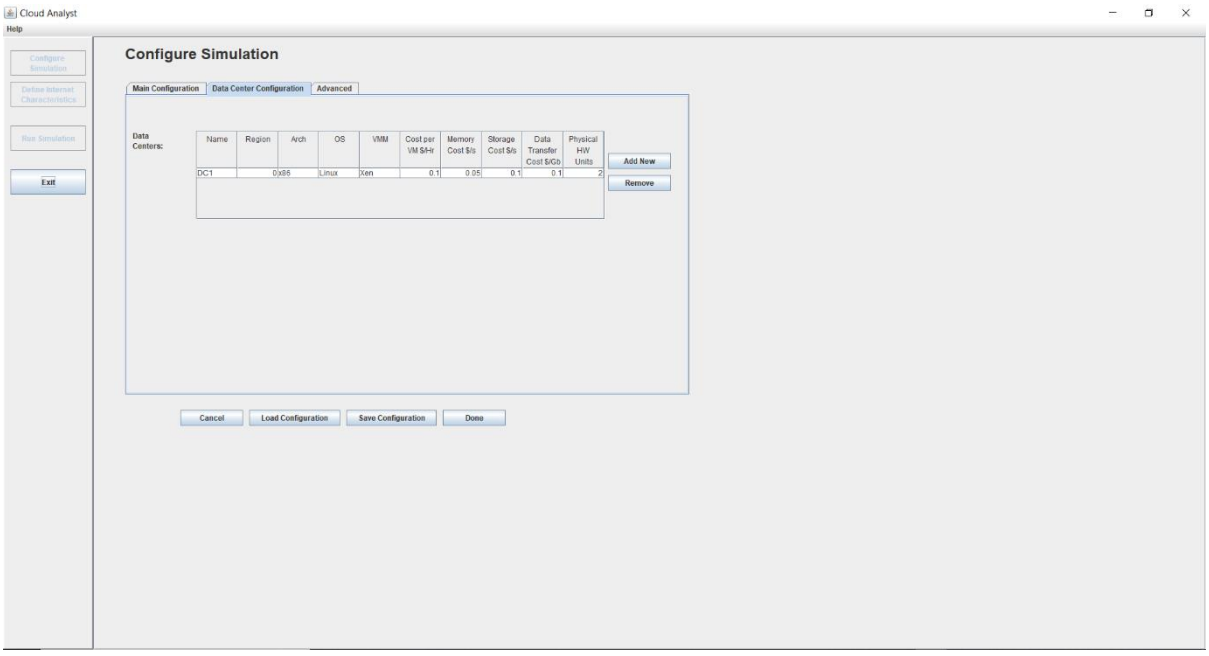
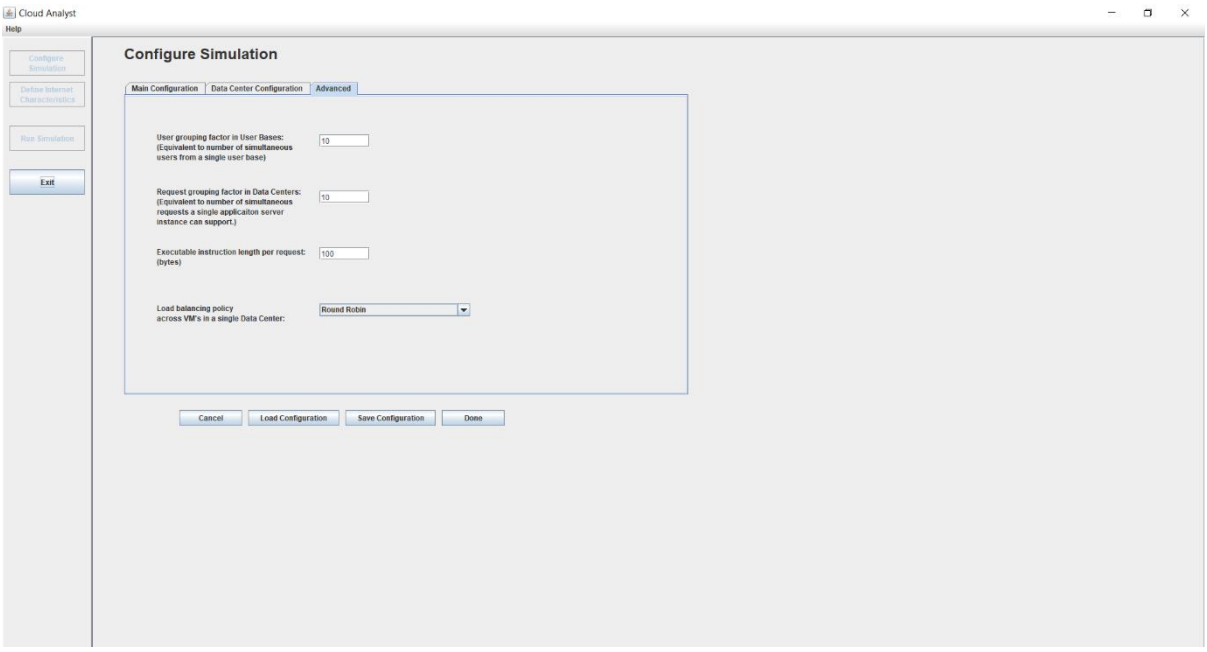


Figure 6 Configure Cloud Analyst

- This is the main configuration page where the simulation duration, user base and application deployment configuration can be edited as per the desired values.

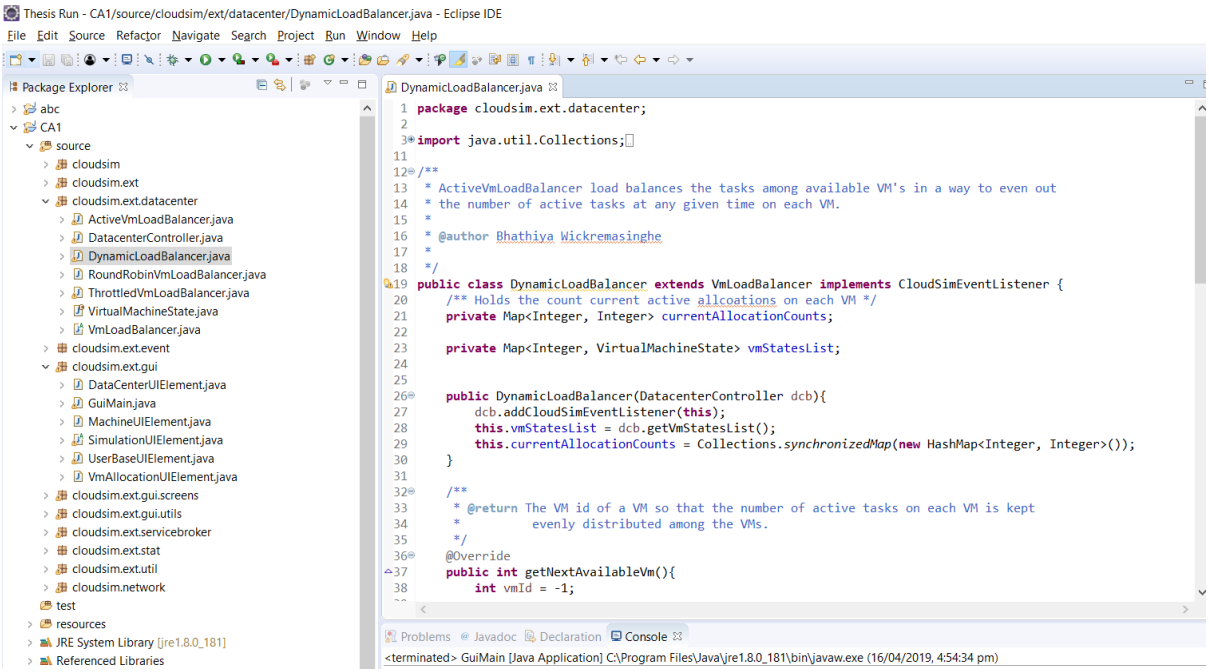


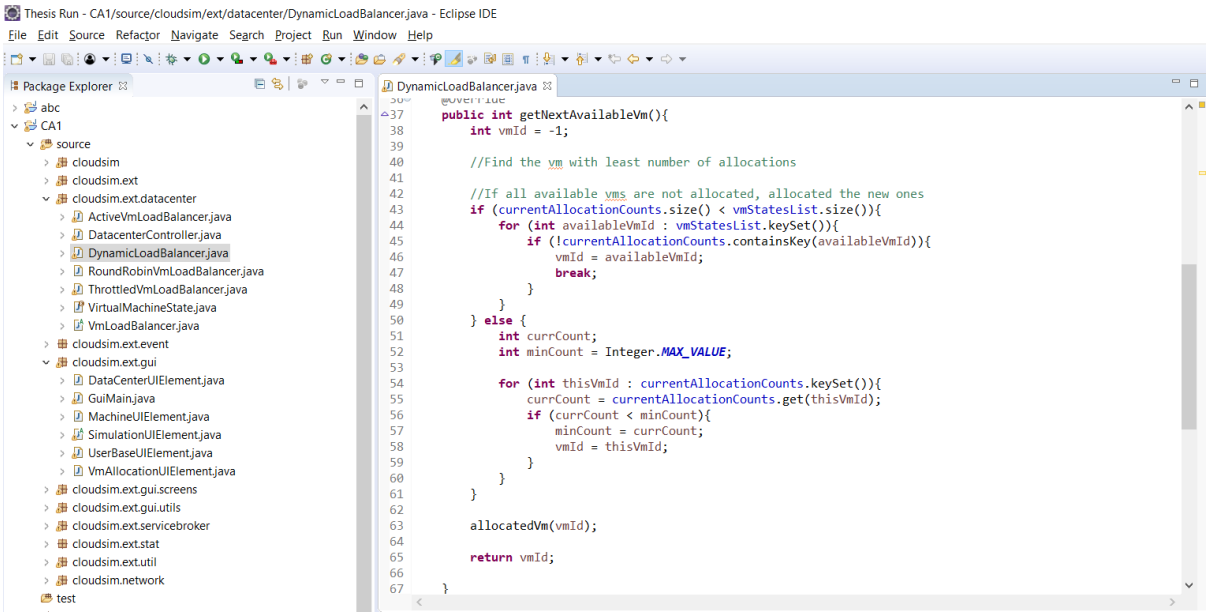
- This is the data centre configuration page where number of data-centres and its individual attributes can be edited and changed as per the desired values.

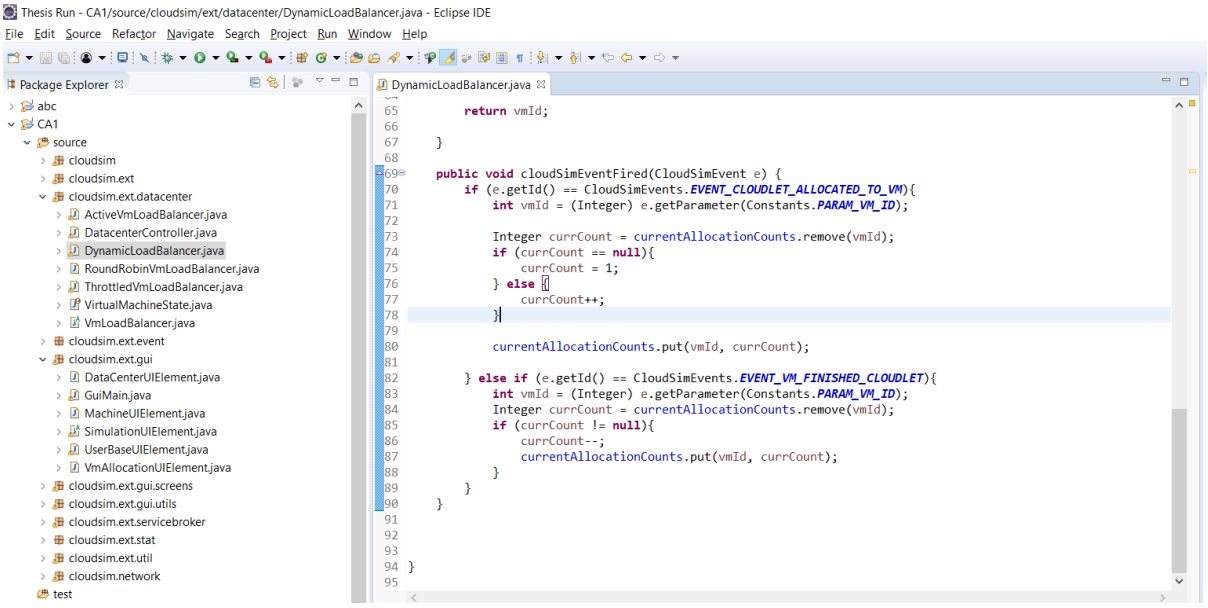


- In the advanced page, the load balancing policy can be selected from the drop box.
- To design new method, the code needs to be written in java.
- After coding, the classes must be defined in the project files and you need to restart the cloud analyst
- After restarting, run the GuiMain file again and in advanced tab, you will be able to see your policy in the drop down button of the load balancing policy.

3. Code for method




Thesis Run - CA1/source/cloudsim/ext/datacenter/DynamicLoadBalancer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
abc
CA1
source
cloudsim
cloudsim.ext
cloudsim.ext.datacenter
ActiveVmLoadBalancer.java
DatacenterController.java
DynamicLoadBalancer.java
RoundRobinVmLoadBalancer.java
ThrottledVmLoadBalancer.java
VirtualMachineState.java
VmLoadBalancer.java
cloudsim.ext.event
cloudsim.ext.gui
DataCenterUIElement.java
GuiMain.java
MachineUIElement.java
SimulationUIElement.java
UserBaseUIElement.java
VmAllocationUIElement.java
cloudsim.ext.gui.screens
cloudsim.ext.gui.utils
cloudsim.ext.servicebroker
cloudsim.ext.stat
cloudsim.ext.util
cloudsim.network
test
DynamicLoadBalancer.java
@Override
public int getNextAvailableVm(){
int vmId = -1;
//Find the vm with least number of allocations
//If all available vms are not allocated, allocated the new ones
if (currentAllocationCounts.size() < vmStatesList.size()){
for (int availableVmId : vmStatesList.keySet()){
if (!currentAllocationCounts.containsKey(availableVmId)){
vmId = availableVmId;
break;
}
}
} else {
int currCount;
int minCount = Integer.MAX_VALUE;
for (int thisVmId : currentAllocationCounts.keySet()){
currCount = currentAllocationCounts.get(thisVmId);
if (currCount < minCount){
minCount = currCount;
vmId = thisVmId;
}
}
}
allocatedVm(vmId);
return vmId;
}


Thesis Run - CA1/source/cloudsim/ext/datacenter/DynamicLoadBalancer.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
abc
CA1
source
cloudsim
cloudsim.ext
cloudsim.ext.datacenter
ActiveVmLoadBalancer.java
DatacenterController.java
DynamicLoadBalancer.java
RoundRobinVmLoadBalancer.java
ThrottledVmLoadBalancer.java
VirtualMachineState.java
VmLoadBalancer.java
cloudsim.ext.event
cloudsim.ext.gui
DataCenterUIElement.java
GuiMain.java
MachineUIElement.java
SimulationUIElement.java
UserBaseUIElement.java
VmAllocationUIElement.java
cloudsim.ext.gui.screens
cloudsim.ext.gui.utils
cloudsim.ext.servicebroker
cloudsim.ext.stat
cloudsim.ext.util
cloudsim.network
test
DynamicLoadBalancer.java
return vmId;
}
public void cloudSimEventFired(CloudSimEvent e) {
if (e.getId() == CloudSimEvents.EVENT_CLOUDLET_ALLOCATED_TO_VM){
int vmId = (Integer) e.getParameter(Constants.PARAM_VM_ID);
Integer currCount = currentAllocationCounts.remove(vmId);
if (currCount == null){
currCount = 1;
} else {
currCount++;
}
currentAllocationCounts.put(vmId, currCount);
} else if (e.getId() == CloudSimEvents.EVENT_VM_FINISHED_CLOUDLET){
int vmId = (Integer) e.getParameter(Constants.PARAM_VM_ID);
Integer currCount = currentAllocationCounts.remove(vmId);
if (currCount != null){
currCount--;
currentAllocationCounts.put(vmId, currCount);
}
}
}
}