CLOUD LOAD BALANCING USING A COMBINATION OF ALGORITHMS FOR THRESHOLD DETECTION AND VM SELECTION

Niladri Sekhar Dey[1] M S K Teja Varma[2] T Nitish[3] M Sai Sunai[4] V Kedareshwar[5]
[1]Assistant Professor
niladri.dey@bvrit.ac.in
[2,3,4,5]Student
[2]18211A1269@bvrit.ac.in, [3]18211A12B1@bvrit.ac.in
[4]18211A1271@bvrit.ac.in, [5]18211A12B7@bvrit.ac.in
[1,2,3,4,5]Department of Information Technology, B V Raju Institute of Technology, Narsapur

Abstract

The Fundamental concept in the current era of cloud computing is to create and share various resource pools comprised of computer resources, primary and secondary storage resources, and a pool of networking resources. These pooled resources are generally structured to be scaled up or scaled down based on access demand and the payment mechanism is also structured as per the usage. Load balancing is the efficient distribution of application traffic across various hosts or Virtual machines. A strategy to curb this overloaded situation is to migrate the VM to less-loaded/ideal host, this is called Virtual Machine Migration. During the VM migration, there is significant degradation in performance due to host shutdown, and inconsistent state issues due to live migration. Also, there is significant energy consumed during this process, which also gives rise to SLA violations. The goal of the paper is to simulate Cloud Load Balancing Using a Combination of Algorithms for Threshold Detection and VM Selection to determine the performance for the combinations using the metrics of Total Migration time, Energy Consumption, SLA Violations, Execution Time, and Number of Virtual Machine Migrations. The performance of the algorithms for each metric is analysed to determine which combination performs the best under that scenario.

Keywords– Cloud Load Balancing;CloudSim; Threshold Detection; Virtual Machine Migration; Robust Local Regression;InterQuartile Range; Local Regression; Maximum Correlation; Minimum Migration Time; Random Selection; Minimum Utilization; Median Absolute Deviation; LRR; LR; MAD; MC; MU; MMT; RS; IQR.

1 Introduction

Cloud computing, or simply cloud, allows the users to create and share various resource pools comprised of computer resources, primary and secondary storage resources, and a pool of networking resources. These pooled resources are generally structured to be scaled up or scaled down based on access demand and the payment mechanism is also structured as per the usage. We see that these cloud computing services are provided in three levels: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). IaaS gives the client the feel of a private server, while the fact is that the service provider provides the client access to a virtual server at a rental subscription. PaaS is the kind of service which allows the developers to build applications and configure other existing SaaS applications by providing them with a cloud framework. SaaS, as the name itself suggests, is an application that is hosted on a cloud for access to the client to use it. The services is made possible by the principle of Virtualization. Virtualization is a technology/principle where the resources are partitioned into multiple virtual hosts having their own virtual environment to function as a

single entity of infrastructure, such entities are called as virtual machines. Based on the type of Virtualization technique used, each Virtual machine has its own operating system and virtual ecosystem to host an application or a piece of application. This allows the hardware resources to be used efficiently and for multiple Virtual machines to run on a single host simultaneously. This causes the Virtual Machines to work under dynamic workload conditions where the resources can be over-utilized or sometimes be under-utilized. To curb such situations of over-utilization of resources and under-utilization of resources, we use the technique of load balancing.

Load balancing is the efficient distribution of application traffic across various hosts or virtual machines. Overload occurs when the resource demands of virtual machines hosting applications exceed the resource consumption threshold.. A threshold value can be provided to each resource, indicating the maximum percentage of usage. An overloaded scenario occurs when the consumption of a resource in a system surpasses its threshold. Assume that threshold for usage is set to be 93% in a host. Whenever the host utilization is over 93%, we say the host is under a load. A strategy to curb this overloaded situation is to migrate the VM to less-loaded/ideal host, this is called Virtual Machine Migration. During this Virtual Machine Migration, certain metrics of energy consumption, total migration time, number of migrations, execution time, and SLA Violations are analysed to determine the performance of the migration process. This is done to identify how the migration should take place to meet the energy or execution time requirements.

2 Foundations of Load Balancing

Load Balancing is the method to reallocate the workload in a distributed environment to ensure that no computing entity is under a load, or idle. Load balancing focuses on parameters like response and execution time, reliability etc to improve the performance of the cloud. In simpler terms, it is an optimization technique to ensure the best potential of the system is experienced. With the extensive access to applications and the internet, the modern-day applications and websites are experiencing high-traffic where they need to serve millions of simultaneous requests by the users and reply with the right kind of information, in a reliable, fast and efficient manner. In order to meet this sudden surge in requests or any scenarios, the applications must be scaled-up efficiently taking into the cost and best practices into consideration. If a single-host goes down, the load balancer ensures that the traffic is redirected to the other hosts. When a new host is added, it ensures that the traffic is redirected to it. Primarily, a load balancer distributes the load efficiently across multiple virtual machines, it ensures high availability and reliability of the system, and provides the flexibility to scale-up and scale-down based on the requirement while taking the cost into consideration for the pay-as-you-go model. VM Migration is one such technique to deal with over-load conditions. In VM migration, an entire VM from one host might be migrated to another host to deal with the overloading scenario. Since the entire VM is being migrated to another host to improve the performance of the system and work under optimal conditions, the major tasks that we come across while Migration is identifying the Virtual Machine that are overloaded or under-utilized, in other words, what is the threshold value that determines whether a particular virtual machine is under-load, this is called as Threshold Detection, and the second task is identifying the Virtual Machine to be migrated to bring the system back to optimal condition for efficient, reliable and cost-effective performance, in other words, Virtual Machine Selection or Consolidation to balance the load among the host. The strategies that are employed for threshold detection and virtual machine consolidation process in this analysis are discussed in the next section. These strategies were designed to take into account the energy usage, migration time, the number of migrations for an optimal scenario, and the execution time for the entire load balancing strategy to terminate successfully.

3 Brief about Algorithms

3.1 Algorithms for threshold detection

3.1.1 Inter-Quartile Range algorithm (IQR)

The Inter-Quartile Range is an estimate of variability, where the data set is divided into quartiles. To obtain the interquartile range, we obtain the difference between the upper quartile and the lower quartile in the dataset. This means that the data set sorted in ascending order is divided into two halves by the median (which is the second quartile), now, the lower quartile is

calculated by finding the median of the first half, and the upper quartile by identifying the median of the second half. In the threshold detection scenario, the data set defines the host utilization metric where the upper and lower quartiles of the sorted set are the medians of the first half and the second half of the data set respectively separated by the median of the dataset. The threshold for determining if the host is overloaded or not is defined as:

$$Threshold, T = 1 - SafetyRange * IQR \quad (1)$$

where, IQR is the InterQuartile Range

3.1.2 Median Absolute Deviation algorithm (MAD)

Median Absolute Deviation is another statistical method to define the threshold for the load balancing. Just like any other deviation where the difference measure is calculated, here, the median is identified and further the median deviation for every element of the data set is calculated by subtracting it with the median. Then, the median absolute deviation is calculated by calculating the mean of the median deviation set. The threshold is defined as:

$$Threshold, T = 1 - SafetyRange * DM \quad (2)$$

where, DM is the Median Absolute Deviation

3.1.3 Local Regression algorithm (LR)

As the name suggests, local regression is used to predict a dependent variable using another independent variable where dependent variable is our research component. To predict our dependent variable, we try to estimate the amount of CPU utilization of the node. Once the value is calculated, we match the value with the actual values of the user nodes and if these values are equal, it implies that the specific user node is overloaded and it needs to be migrated or resources should be extended. Otherwise, the node can be considered as a safe node. The two main components in creating a equation which could predict the overloading value of a node are the CPU utilisation of that node and the Load pattern of that node. These two components are used to calculate the CPU utilization weight function and load pattern weight functions which play a key role in creating the estimation function. Using these weight functions, we determine the threshold value that is used to identify if the host is overloaded or not.

3.1.4 Robust Local Regression Algorithm (LRR)

The key difference between the local regression method and robust local regression method is the errors that come across in the LR are reduced by involving a combined weight function which is created using both load pattern and CPU utilization ie.,in the local regression method, a weight function is created using CPU utilization values of the node, in the same way, another weight function is created using the load patterns of the node. But, in robust local regression method, a weight function is generated by the sum of the two weight functions involving the load pattern and CPU utilization values. This is used as the final equation to estimate the value which can be used to compare the actual values of node and decide if the host is under a load or not.

3.1.5 Static Threshold Method (THR)

In the static threshold method, certain assumptions are made about the load and the host utilization to define or computer the threshold values to determine the virtual machines and hosts that are overloaded.

3.2 Algorithms for Virtual Machine Consolidation

3.2.1 Maximum Correlation (MC)

In this method of virtual machine consolidation, we use the principle of maximum correlation where the relation between two parameters is computed using a domain function. In the Virtual Machine Consolidation Scenario, all the overloaded virtual machines are accepted as input. We define a two-dimensional matrix of the Available and Utilized resources of the overloaded virtual machines. Then, we find the transpose of the matrix. Now, we apply the principle of correlation to the two matrices and identify the VMs with the highest correlation.

3.2.2 Minimum Migration Time (MMT)

While the Migration time is the sum of the shutdown time, start-up time and the transfer time, this migration time principle is used for determining the VMs that should be migrated to balance the load. In majority of the scenarios, it was observed that the many virtual machines or hosts were in similar load scenarios so, identifying the specific least loaded node was an expensive task, but determining the virtual machine with least migration time was economical and energy-efficient. Hence, the

principle to compute the VM selection that takes minimum migration time is used.

3.2.3 Random Selection (RS)

The key term that is implied here is random. The virtual machine that is selected in random has no definite selection criteria or does not follow any pattern of selection. This random selection plays a key role in large problem space where the time complexity for the selection of best-case scenario is very less, however, the accuracy is a questionable concern in this method of selection. This principle is used in our load balancing scenario where every VM machine in load is migrated to a lesser loaded node and if the load is balanced, the migration process is terminated otherwise, an virtual machine is selected in random to migrate.

3.2.4 Minimum Utilization (MU)

The primary goal of Minimum utilization method is to produce lower overhead for job processing. Here, this policy ensures that the less overloaded Virtual machines is migrated to hosts that are moderately loaded to balance the load keeping the goal to reduce consumption of energy.

All the possible combinations of algorithm include LR MC, LR MMT, IQR MC, MAD MC, MAD MMT, LRR MU, LRR RS, THR MC, THR MMT, THR MU, IQR MMT, IQR MU, IQR RS, LR MU, MAD MU, MAD RS, LRR MC, LRR MMT, and THR RS.

4 Results

4.1 Energy Consumption

Energy Consumption is the amount of power utilized for performing some work in unit amount of time. Here, energy consumption refers to the power usage to perform the Virtual Machine Migration. It is one of the primary factors that is considered when analyzing and determining the strategies used for VM migration. Power consumption has become a critical factor with the rise of green computing and the need to reduce the consumption of power by several datacenters. The global datacenter consumes about 3% of world's power production, this further leads to increased CO2.

The Table displays the results of the energy consumption analysis (in sorted order of lower to higher energy consumption) when the load balancing is simulated with each algorithm (in kWh) against the 10 datasets and The figure visualizes the results graphically. The line graph represents the average energy consumption for each algorithm.

On analyzing the Energy Consumption metrics, It was observed that for algorithms robust local regression or local regression in combination with random selection or maximum correlation lies in the top 20% performance range. Here, the average of these combinations was observed to be 34.3215 kWh compared to the overall average of 41.0335 kWh is 16.357% less energy consumption. Optimally, the combination of robust local regression and random selection consumed 34.218 kWh which is 16.61% less energy consumed compared to the average energy consumption. Also, the combination of the Inter-quartile range with Minimum Utilization performed the worst in the analysis consuming 20% more energy than the average. The algorithm Inter-quartile range in combination with Minimum-Migration Time or Minimum Utilization lies in the bottom 20% performance range consuming 16.72% more energy than the average energy consumption.

	Dataset									
Experiment Name	20110303	20110306	20110309	20110322	20110325	20110403	20110409	20110411	20110412	20110420
lrr rs 1.2	33.79	34.34	34.08	34.38	34.44	33.93	34.51	34.29	34.14	34.28
lrr mc 1.2	34.35	34.35	34.35	34.35	34.35	34.35	34.35	34.35	34.35	34.35
lr mc 1.2	34.35	34.35	34.35	34.35	34.35	34.35	34.35	34.35	34.35	34.35
lr rs 1.2	34.41	34.12	34.64	34.37	34.13	34.04	34.22	34.44	34.59	34.72
lrr mmt 1.2	35.37	35.37	35.37	35.37	35.37	35.37	35.37	35.37	35.37	35.37
lr mmt 1.2	35.37	35.37	35.37	35.37	35.37	35.37	35.37	35.37	35.37	35.37
lrr mu 1.2	35.38	35.38	35.38	35.38	35.38	35.38	35.38	35.38	35.38	35.38
lr mu 1.2	35.38	35.38	35.38	35.38	35.38	35.38	35.38	35.38	35.38	35.38
thr mc 0.8	40.85	40.85	40.85	40.85	40.85	40.85	40.85	40.85	40.85	40.85
thr rs 0.8	41.22	41.2	41.1	41.23	41.21	41.31	40.97	41.08	41.36	41.06
thr mmt 0.8	41.81	41.81	41.81	41.81	41.81	41.81	41.81	41.81	41.81	41.81
thr mu 0.8	44.08	44.08	44.08	44.08	44.08	44.08	44.08	44.08	44.08	44.08
mad rs 2.5	44.79	45.18	45.26	44.5	45.08	44.6	45	44.85	45.04	44.89
mad mc 2.5	44.99	44.99	44.99	44.99	44.99	44.99	44.99	44.99	44.99	44.99
mad mmt 2.5	45.61	45.61	45.61	45.61	45.61	45.61	45.61	45.61	45.61	45.61
iqr mc 1.5	46.86	46.86	46.86	46.86	46.86	46.86	46.86	46.86	46.86	46.86
iqr rs 1.5	46.96	47.08	46.7	47.17	46.95	47.31	47.13	46.88	47.28	47.15
mad mu 2.5	47.36	47.36	47.36	47.36	47.36	47.36	47.36	47.36	47.36	47.36
iqr mmt 1.5	47.85	47.85	47.85	47.85	47.85	47.85	47.85	47.85	47.85	47.85
iqr mu 1.5	49.32	49.32	49.32	49.32	49.32	49.32	49.32	49.32	49.32	49.32

Table 1: Energy Consumption(in kWh) Analysis

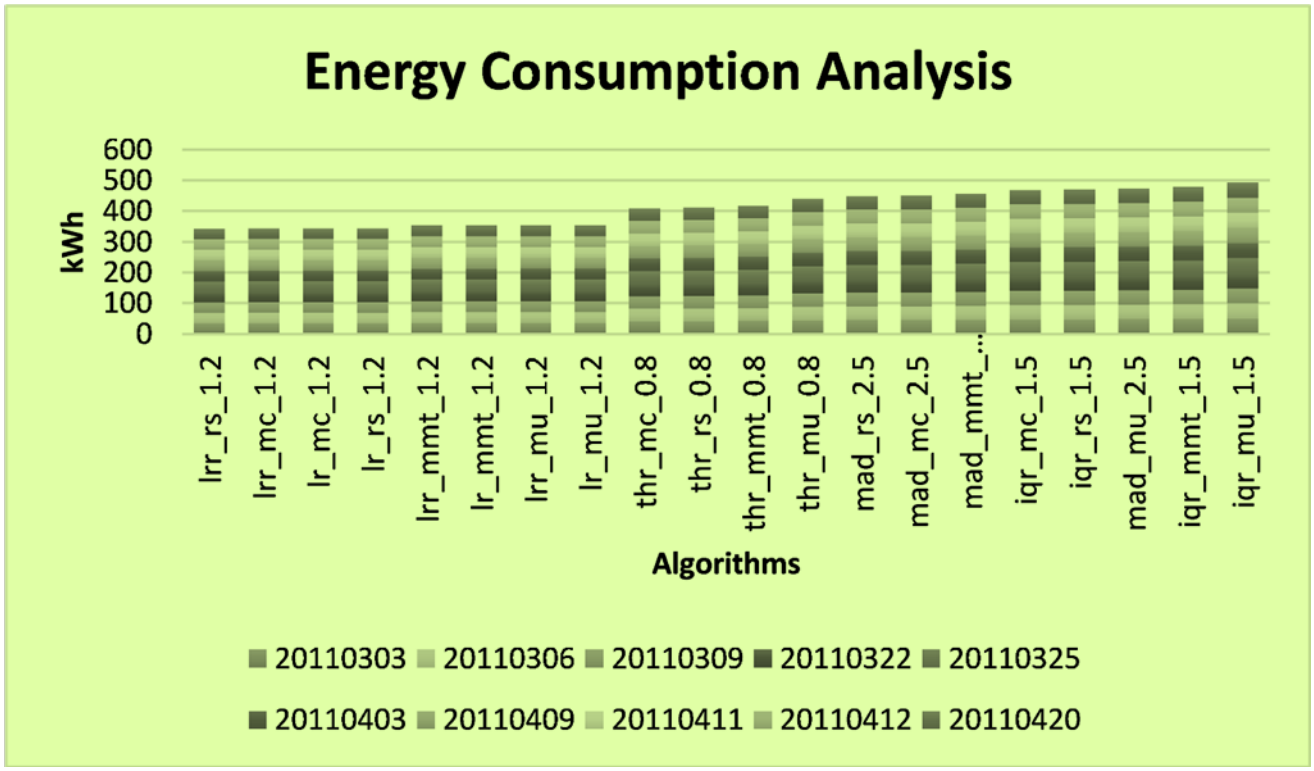

Figure 1: Energy Consumption Analysis

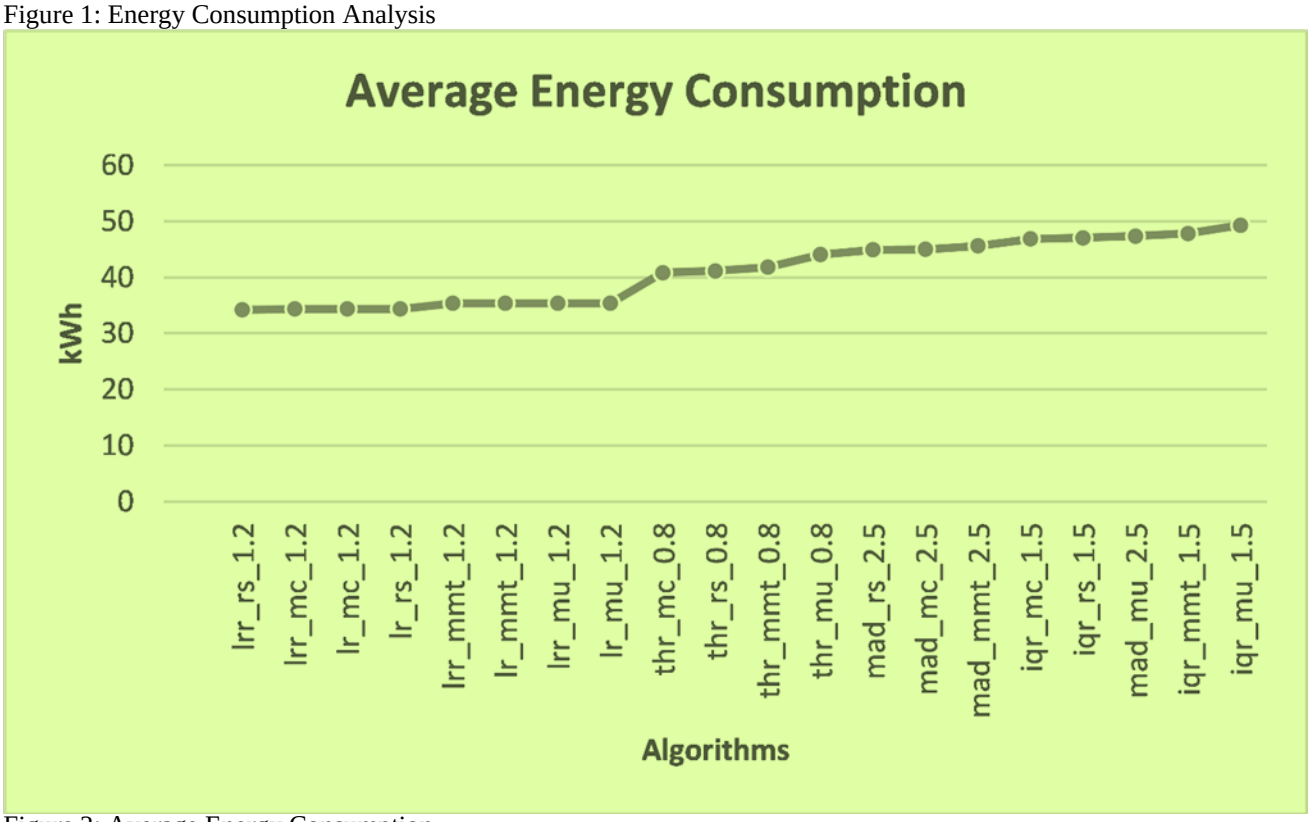

Figure 2: Average Energy Consumption

4.2 Number of VM Migrations

The Number of Virtual Machine Migrations determine the downtime of the application when the live migration is being performed. It is imperative to analyze and calculate the number of migrations as it leads to host shutdown and increase in the downtime, this further contributes to the decrease in responsiveness. As we understand that SLA satisfiability is determined by the responsiveness, the number of migrations contribute to either better satisfiability or violations. The number of Migrations are calculated by finding the correlation between the list of overloaded virtual machines and the virtual machines that are migrated.

The Table displays the results of the VM Migrations Analysis (in sorted order of lower to higher number of VM Migrations) when the load balancing is simulated with each algorithm against the 10 datasets and The figure visualizes the results graphically. The line graph

represents the average number of Migrations for each algorithm.

The results from the VM Migrations analysis showed the average number of migrations to be 4067.195 while the top 40% of the observations lie below the average. It was observed these top 40% include robust local regression methods and local regression methods for threshold detection in combination with the four VM selection algorithms. Here, the robust local regression method and local regression method in combination with maximum correlation for VM Selection performed the best with 2203 migrations, which is 45.83% less than the average. The top 20% observations include the combinations of robust local regression method and local regression method for threshold detection with Random Selection and Maximum Correlation with an average of 2245.7 migrations, which is 44.785% less than the average number of VM Migrations. Also, the ninth observation: static threshold method with maximum correlation, compared to the 8th observation: local regression method with minimum migration time, which is the last below-average observation, showed a 34.6% increase in the number of VM Migrations. Also, The algorithm Minimum Utilization with static threshold method or Inter-Quartile Range or Median Absolute Deviation, and the combination of Inter-Quartile Range and Minimum Migration Time performed at the bottom 20% where the average of the bottom 20% is 37.21% more than the Average VM Migrations, where the combination of Inter-Quartile Range with Minimum utilization performed the worst with 5789 migrations, i.e. 42.33% more than the average number of migrations.

	Dataset									
Experiment Name	**20110303**	**20110306**	**20110309**	**20110322**	**20110325**	**20110403**	**20110409**	**20110411**	**20110412**	**20110420**
lrr rs 1.2	2086	2409	2131	2417	2335	2336	2204	2359	2306	2250
lrr mc 1.2	2203	2203	2203	2203	2203	2203	2203	2203	2203	2203
lr mc 1.2	2203	2203	2203	2203	2203	2203	2203	2203	2203	2203
lr rs 1.2	2366	2245	2313	2247	2317	2114	2333	2306	2240	2454
lrr mu 1.2	2808	2808	2808	2808	2808	2808	2808	2808	2808	2808
lr mu 1.2	2808	2808	2808	2808	2808	2808	2808	2808	2808	2808
lrr mmt 1.2	2872	2872	2872	2872	2872	2872	2872	2872	2872	2872
lr mmt 1.2	2872	2872	2872	2872	2872	2872	2872	2872	2872	2872
thr mc 0.8	4392	4392	4392	4392	4392	4392	4392	4392	4392	4392
thr rs 0.8	4532	4529	4407	4399	4481	4568	4389	4504	4471	4433
mad mc 2.5	4778	4778	4778	4778	4778	4778	4778	4778	4778	4778
mad rs 2.5	4831	4871	4853	4753	4810	4786	4685	4875	4886	4797
thr mmt 0.8	4839	4839	4839	4839	4839	4839	4839	4839	4839	4839
iqr rs 1.5	5066	4999	5031	4990	5094	5057	4999	4994	5079	5022
iqr mc 1.5	5085	5085	5085	5085	5085	5085	5085	5085	5085	5085
mad mmt 2.5	5265	5265	5265	5265	5265	5265	5265	5265	5265	5265
thr mu 0.8	5404	5404	5404	5404	5404	5404	5404	5404	5404	5404
iqr mmt 1.5	5502	5502	5502	5502	5502	5502	5502	5502	5502	5502
mad mu 2.5	5628	5628	5628	5628	5628	5628	5628	5628	5628	5628
iqr mu 1.5	5789	5789	5789	5789	5789	5789	5789	5789	5789	5789

Table 2: VM Migration Analysis

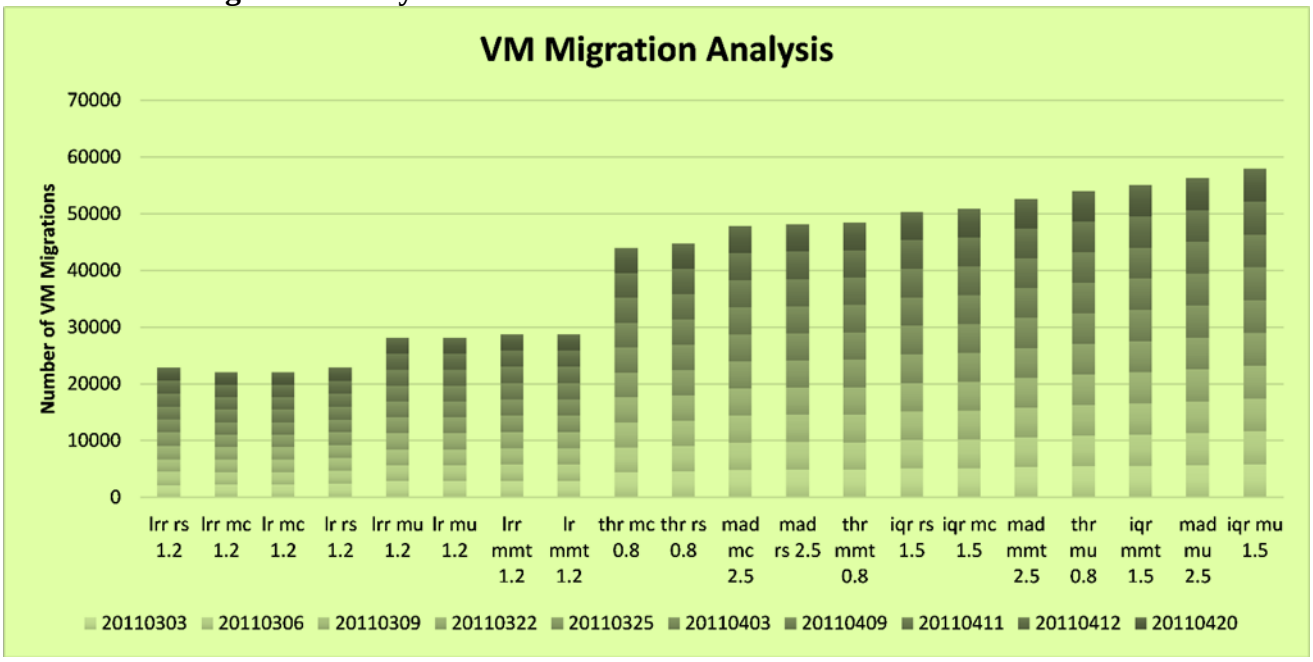

Figure 3: VM Migration Analysis

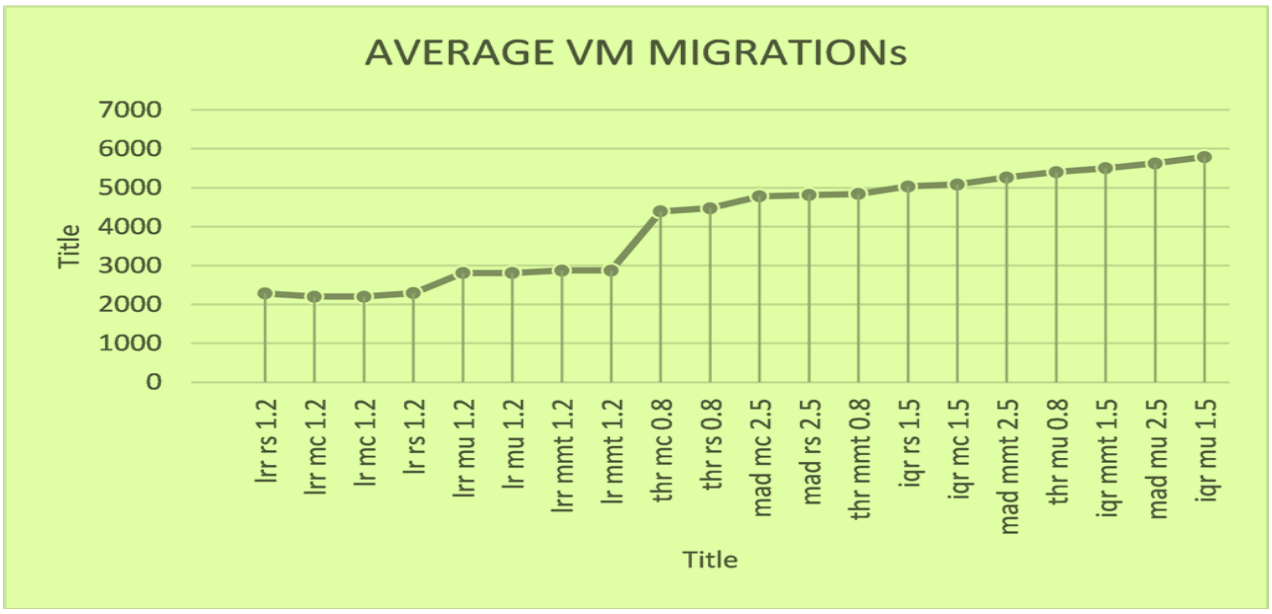

Figure 4: Average VM Migrations

4.3 SLA Performance Degradation due to Migration

The Service Level Agreement measure determines the responsiveness of the system. The agreement between the service provider and client requires the provider to adhere to certain quality of service expected by the client. In general, the client expects higher SLA satisfiability and this is hampered by the migration. However, the migration also helps in reducing the wait time and improves the behavior of the system when provided with sufficient resources, thus promoting better SLA Satisfiability. The SLA violation is calculated as the difference between the SLA measure agreed by the provider and client and the sum of the virtual machine shutdown, transfer and startup time.

The Table displays the results of the SLA Violations analysis (in sorted order of lower to higher percentage of SLA Violations) when the load balancing is simulated with each algorithm against the 10 datasets and The figure visualizes the results graphically. The column graph represents the average percentage of SLA Violations for each algorithm.

While the average SLA performance degradation was observed to be 0.2075%, the top 4 observations which include the robust local regression method and local regression method for threshold detection in combination with Minimum Migration time and Minimum Utilization for VM Selection, showed a constant observation of 0.13% which is 0.07% less than the average. Also, there is a signification increase of 0.88% in the performance degradation from the eighth(the last below-average observation)and ninth observations, which are the local regression method with random selection, and Inter-Quartile Range with Minimum Migration time respectively. The bottom 15% includes the Static Threshold Method for Threshold Detection with Maximum Correlation, Random Selection, and Minimum Utilization for VM selection with an average of 0.27467% which is 0.07% more than the average.

	Dataset									
Experiment Name	20110303	20110306	20110309	20110322	20110325	20110403	20110409	20110411	20110412	20110420
lrr_mmt_1.2	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
lrr_mu_1.2	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
lrr_rs_1.2	0.13	0.15	0.13	0.14	0.14	0.14	0.14	0.14	0.14	0.14
lr_mmt_1.2	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
lr_mu_1.2	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13
lrr_mc_1.2	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14
lr_mc_1.2	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14	0.14
lr_rs_1.2	0.14	0.14	0.14	0.14	0.15	0.13	0.15	0.14	0.14	0.15
iqr_mmt_1.5	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23
mad_mmt_2.5	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23
thr_mmt_0.8	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23	0.23
iqr_rs_1.5	0.25	0.25	0.26	0.25	0.25	0.26	0.26	0.25	0.26	0.25
iqr_mc_1.5	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
iqr_mu_1.5	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
mad_mc_2.5	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
mad_mu_2.5	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
mad_rs_2.5	0.26	0.25	0.26	0.25	0.25	0.25	0.25	0.27	0.26	0.26
thr_mc_0.8	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
thr_rs_0.8	0.27	0.29	0.27	0.27	0.27	0.27	0.28	0.28	0.27	0.27
thr_mu_0.8	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28

Table 3: SLA Violations(in %) Analysis

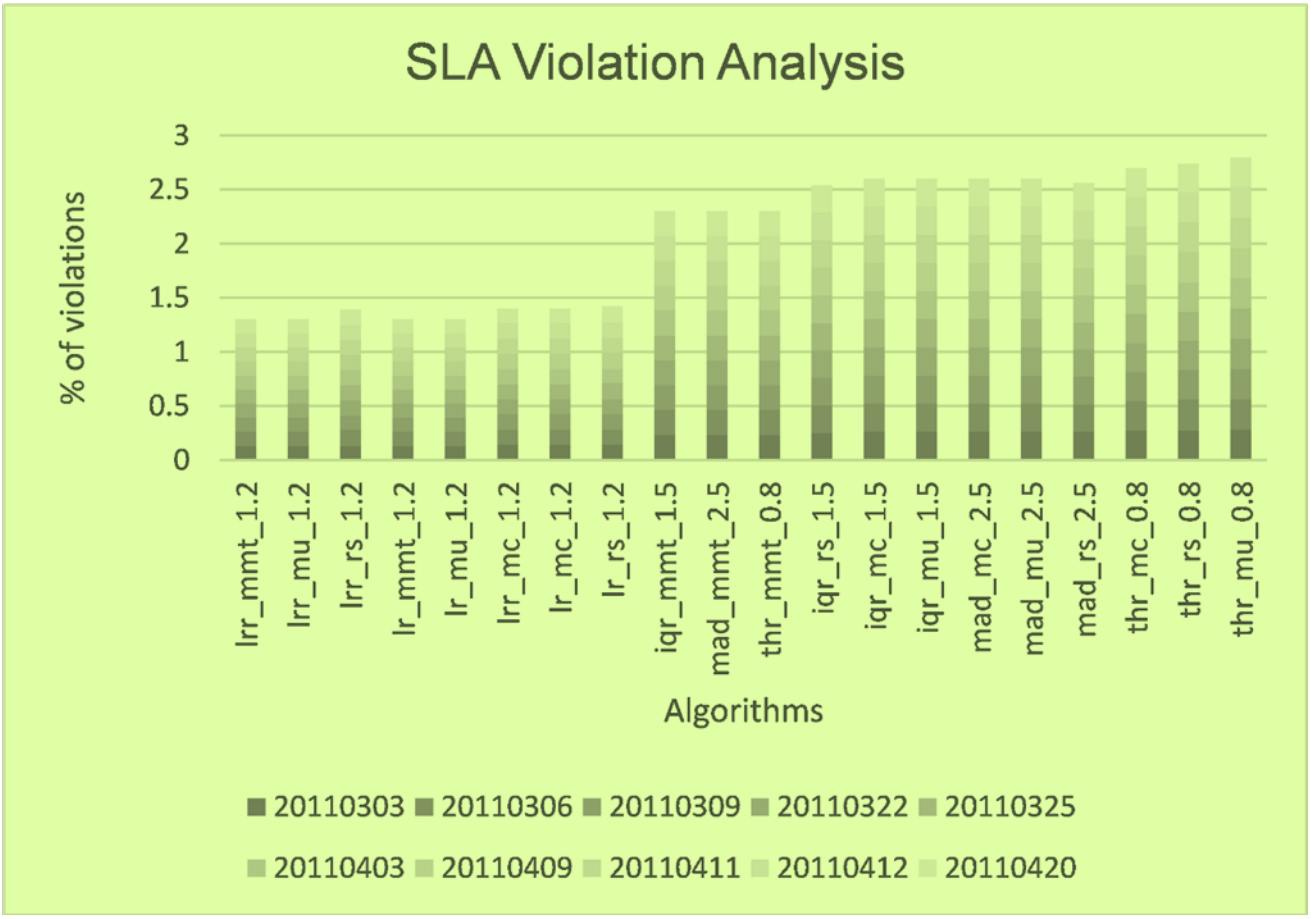

Figure 5: SLA Violations(in %) Analysis.

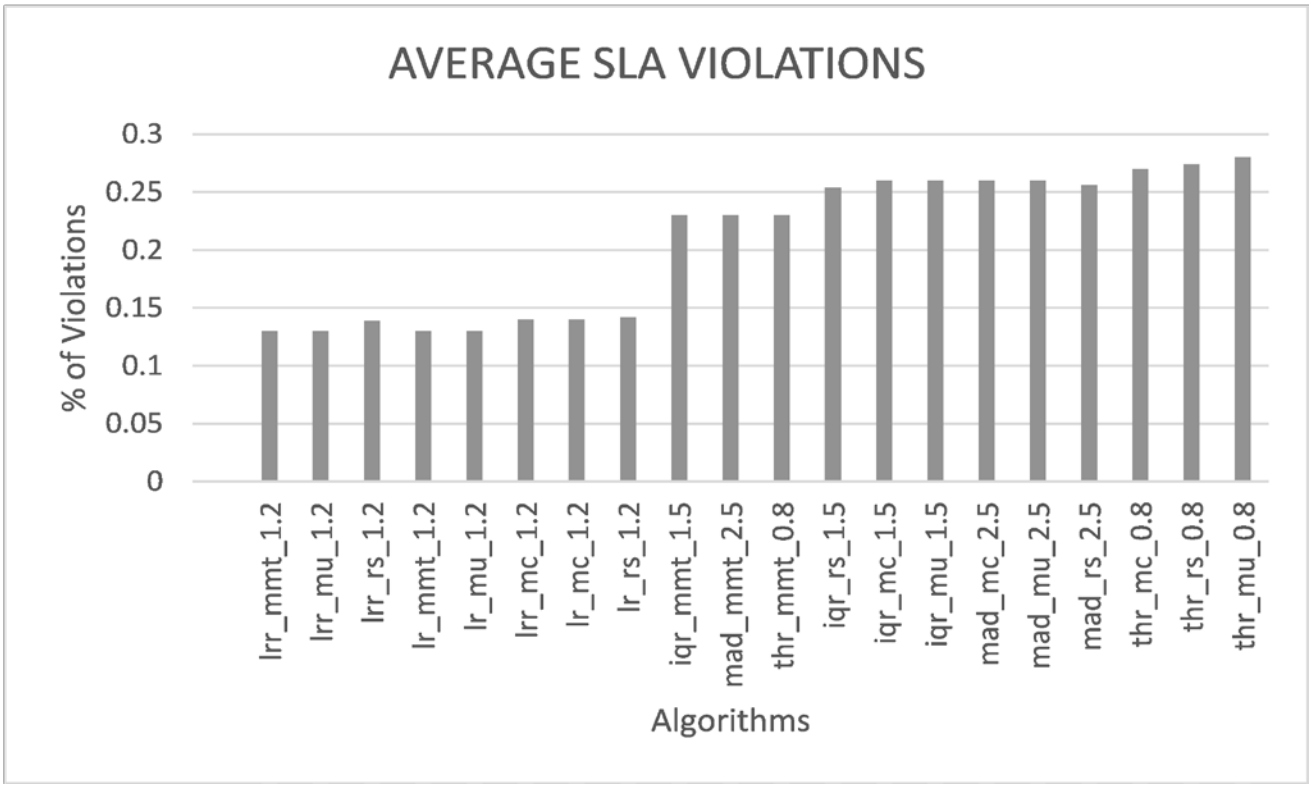

Figure 6: Average SLA Violations(in %)

4.4 Meantime before VM Migration

The meantime before the virtual machine Migration is the time taken to identify the loaded VM to be migrated from the mean time of host shutdown. This mean time denotes the downtime experienced during the Migration and affects the recovery of the system.

The Table displays the results of Mean time (in sec) before VM Migration analysis (in sorted order of lower to higher seconds of time) when the load balancing is simulated with each algorithm against the 10 datasets and The figure visualizes the results graphically.The line graph represents the average mean time before VM Migration for each algorithm.

On analyzing the Meantime before VM Migration metrics, it is observed the top 25% of observations lie below the total average of 19.4615 seconds. This top 25% includes the Minimum Migration time for VM Selection in combination with the five threshold detection methods, where the local regression method and robust local regression method for threshold detection performed the best at 16.6 seconds with 2.86 seconds less than the average. Also, the bottom 75% of observations have an average of 20.295 seconds which is 0.8332 seconds less than the average.

	Dataset									
Experiment Name	20110303	20110306	20110309	20110322	20110325	20110403	20110409	20110411	20110412	20110420
lrr_mmt_1.2	16.6	16.6	16.6	16.6	16.6	16.6	16.6	16.6	16.6	16.6
lr_mmt_1.2	16.6	16.6	16.6	16.6	16.6	16.6	16.6	16.6	16.6	16.6
thr_mmt_0.8	16.82	16.82	16.82	16.82	16.82	16.82	16.82	16.82	16.82	16.82
mad_mmt_2.5	17.17	17.17	17.17	17.17	17.17	17.17	17.17	17.17	17.17	17.17
iqr_mmt_1.5	17.62	17.62	17.62	17.62	17.62	17.62	17.62	17.62	17.62	17.62
thr_rs_0.8	20.05	20.45	20.41	20.17	20.26	20.43	20.6	20.2	20.26	20.29
lrr_mu_1.2	20.06	20.06	20.06	20.06	20.06	20.06	20.06	20.06	20.06	20.06
lr_mu_1.2	20.06	20.06	20.06	20.06	20.06	20.06	20.06	20.06	20.06	20.06
mad_mu_2.5	20.18	20.18	20.18	20.18	20.18	20.18	20.18	20.18	20.18	20.18
mad_rs_2.5	20.21	20.15	20.29	20.22	20.17	20.32	20.24	20.3	20.14	20.24
thr_mu_0.8	20.23	20.23	20.23	20.23	20.23	20.23	20.23	20.23	20.23	20.23
iqr_mc_1.5	20.33	20.33	20.33	20.33	20.33	20.33	20.33	20.33	20.33	20.33
lrr_mc_1.2	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35
lr_mc_1.2	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35
mad_mc_2.5	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35	20.35
iqr_mu_1.5	20.38	20.38	20.38	20.38	20.38	20.38	20.38	20.38	20.38	20.38
lrr_rs_1.2	20.38	20.43	20.25	20.17	20.2	20.12	20.52	20.56	20.59	20.44
lr_rs_1.2	20.38	20.61	20.4	20.38	20.4	20.42	20.6	20.4	20.34	20.46
iqr_rs_1.5	20.47	20.25	20.42	20.15	20.27	20.26	20.4	20.11	20.38	20.44
thr_mc_0.8	20.47	20.47	20.47	20.47	20.47	20.47	20.47	20.47	20.47	20.47

Table 4: Meantime before Migration (in sec) Analysis

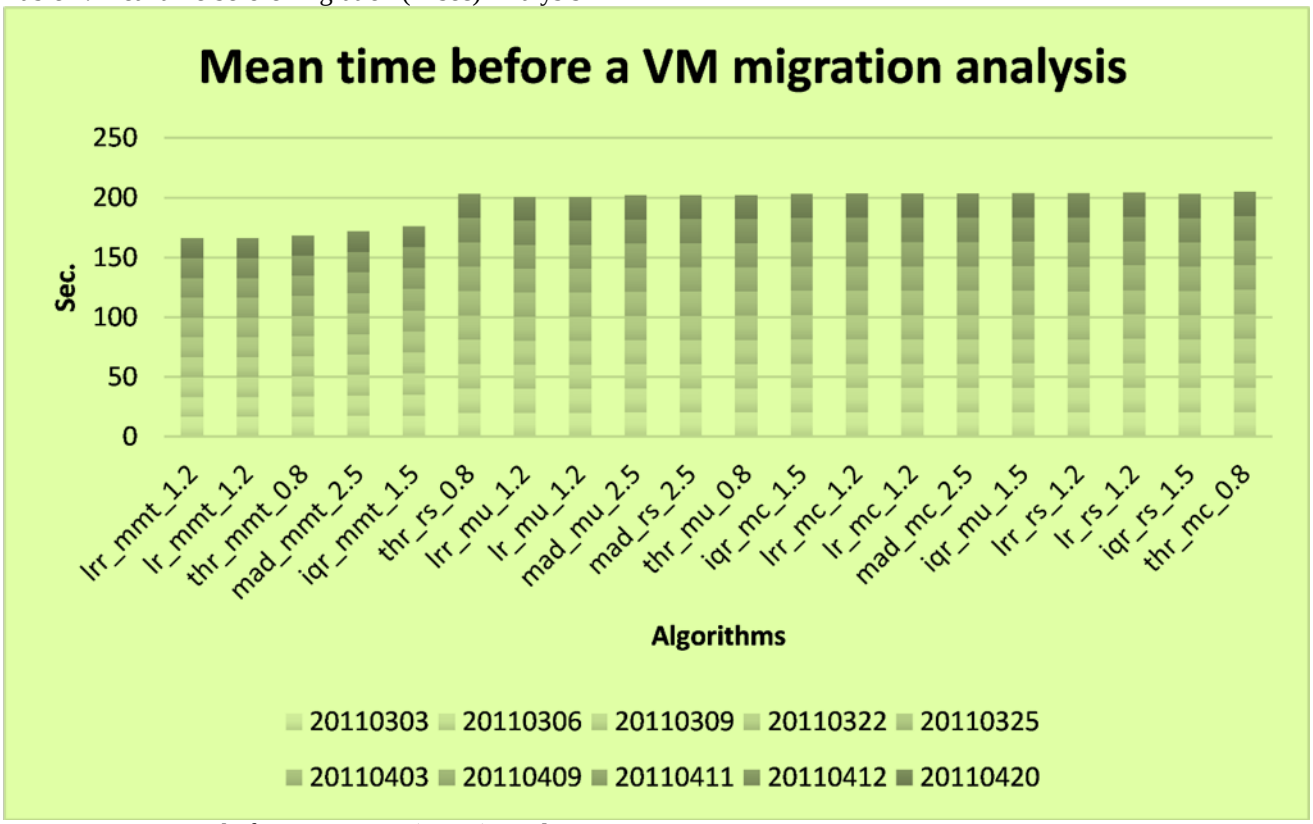

Figure 7: Meantime before Migration (in sec) Analysis

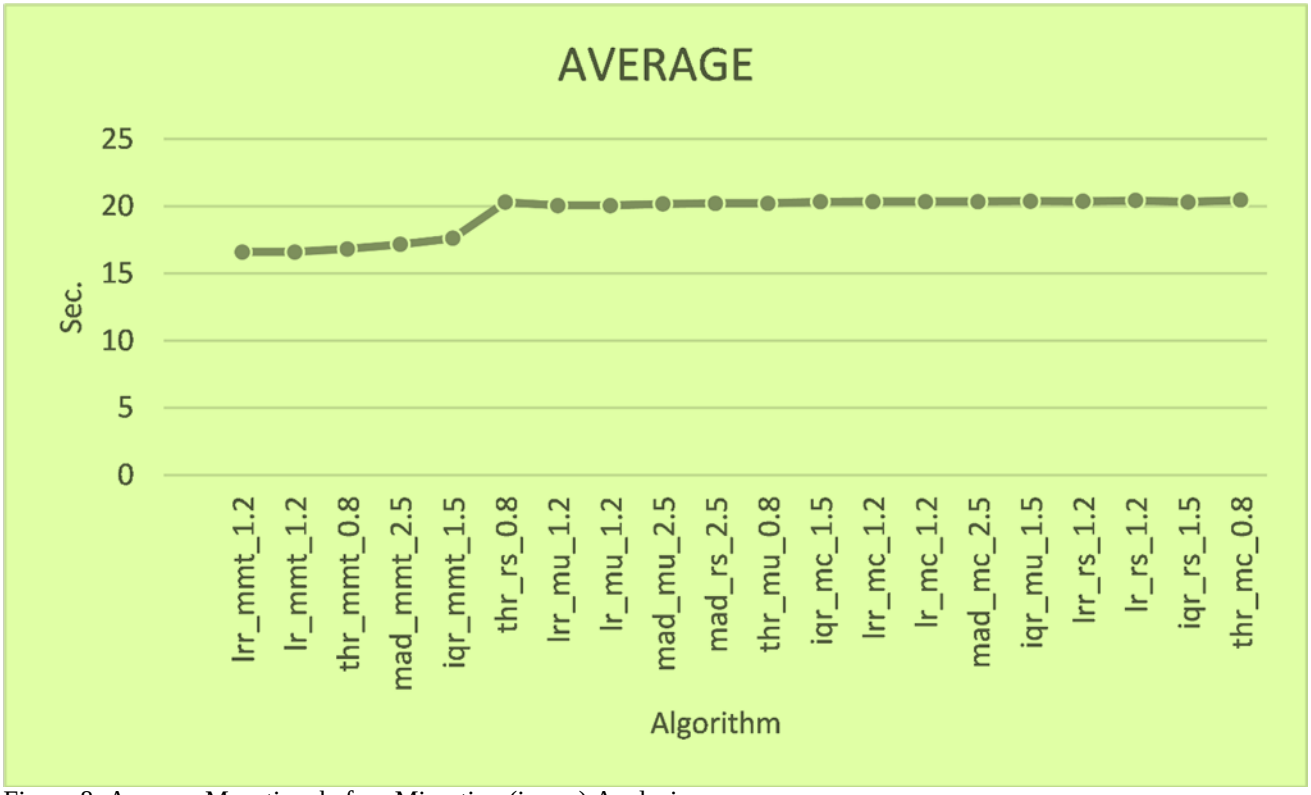

Figure 8: Average Meantime before Migration (in sec) Analysis

4.5 Execution Time

The Execution time denotes the time taken for the entire experiment to finish for that particular algorithm combination against the particular dataset. The Execution time, in general, represent the time taken for the load balancing to successful finish in the system.

The Table displays the results of the Execution time(sorted in the order of lowest to highest execution time)(in sec) when the load balancing is simulated with each algorithm against the 10 datasets and The figure visualizes the results graphically. The line graph represents the average execution time for each algorithm.

The top 20% of observations have an average of 0.001 seconds which is 0.0005 seconds less than the total average of 0.0015 seconds. These top 20% observations include the local regression method and static threshold method for threshold detection in combination with random selection and minimum migration time for VM selection, while the combination of static threshold method and minimum migration time performs the best with an execution time of 0.000977 seconds which is 36% less than the average execution time. Also, the bottom 40% of observations, with an average of 0.0022 seconds, lie below the average execution time, where the combination of inter-quartile range and maximum correlation performed the worst with 0.002694 seconds which is 74% more than the average execution time.

	Dataset									
Experiment Name	**20110303**	**20110306**	**20110309**	**20110322**	**20110325**	**20110403**	**20110409**	**20110411**	**20110412**	**20110420**
thr_rs_0.8	0.00082	0.0008	0.00091	0.00081	0.00078	0.00211	0.00197	0.00083	0.00084	0.00043
lr_rs_1.2	0.00084	0.00069	0.00085	0.00088	0.0008	0.00093	0.00216	0.00129	0.00091	0.001
thr_mmt_0.8	0.00085	0.00046	0.00094	0.00089	0.00088	0.00081	0.00201	0.0008	0.00119	0.00094
lr_mu_1.2	0.00089	0.00096	0.00089	0.00103	0.00098	0.001	0.0024	0.00086	0.00091	0.0009
lr_mmt_1.2	0.0009	0.00079	0.00092	0.00103	0.001	0.001	0.00236	0.00076	0.00092	0.00075
lrr_rs_1.2	0.00093	0.00099	0.00095	0.00101	0.00097	0.001	0.00238	0.00094	0.001	0.00093
lr_mc_1.2	0.00094	0.00133	0.001	0.00116	0.00117	0.00105	0.00248	0.00085	0.00105	0.00099
thr_mu_0.8	0.00095	0.0008	0.00051	0.00099	0.00101	0.00193	0.00237	0.00095	0.00086	0.00044
thr_mc_0.8	0.001	0.00147	0.00119	0.00114	0.001	0.00105	0.00244	0.00096	0.00081	0.00076
lrr_mmt_1.2	0.00105	0.00097	0.00107	0.0012	0.00114	0.00112	0.00276	0.00108	0.0011	0.00102
lrr_mu_1.2	0.00109	0.00062	0.00102	0.00118	0.00106	0.00107	0.00274	0.00106	0.00109	0.00145
lrr_mc_1.2	0.00116	0.00108	0.00112	0.00132	0.00123	0.00121	0.003	0.00086	0.00123	0.00078
iqr_rs_1.5	0.00143	0.00143	0.00137	0.00157	0.00146	0.00161	0.00383	0.00154	0.00157	0.00134
iqr_mmt_1.5	0.00148	0.00129	0.00155	0.00177	0.0017	0.00176	0.0043	0.00163	0.00166	0.00159
iqr_mu_1.5	0.00157	0.00166	0.0016	0.00181	0.00175	0.0019	0.0045	0.00155	0.00179	0.00188
mad_rs_2.5	0.00176	0.00125	0.0017	0.00195	0.00181	0.00179	0.00439	0.00201	0.00168	0.0013
mad_mmt_2.5	0.00197	0.00196	0.002	0.002	0.00204	0.00202	0.00496	0.00188	0.00226	0.00192
iqr_mc_1.5	0.00205	0.00208	0.00247	0.00265	0.00241	0.00274	0.00541	0.00247	0.00248	0.00218
mad_mc_2.5	0.00206	0.00202	0.00184	0.00214	0.00209	0.00206	0.00511	0.00193	0.0015	0.00218
mad_mu_2.5	0.00215	0.00226	0.00203	0.0023	0.00218	0.00227	0.0055	0.00227	0.00203	0.00246

Table 5: Execution time(in sec) Analysis

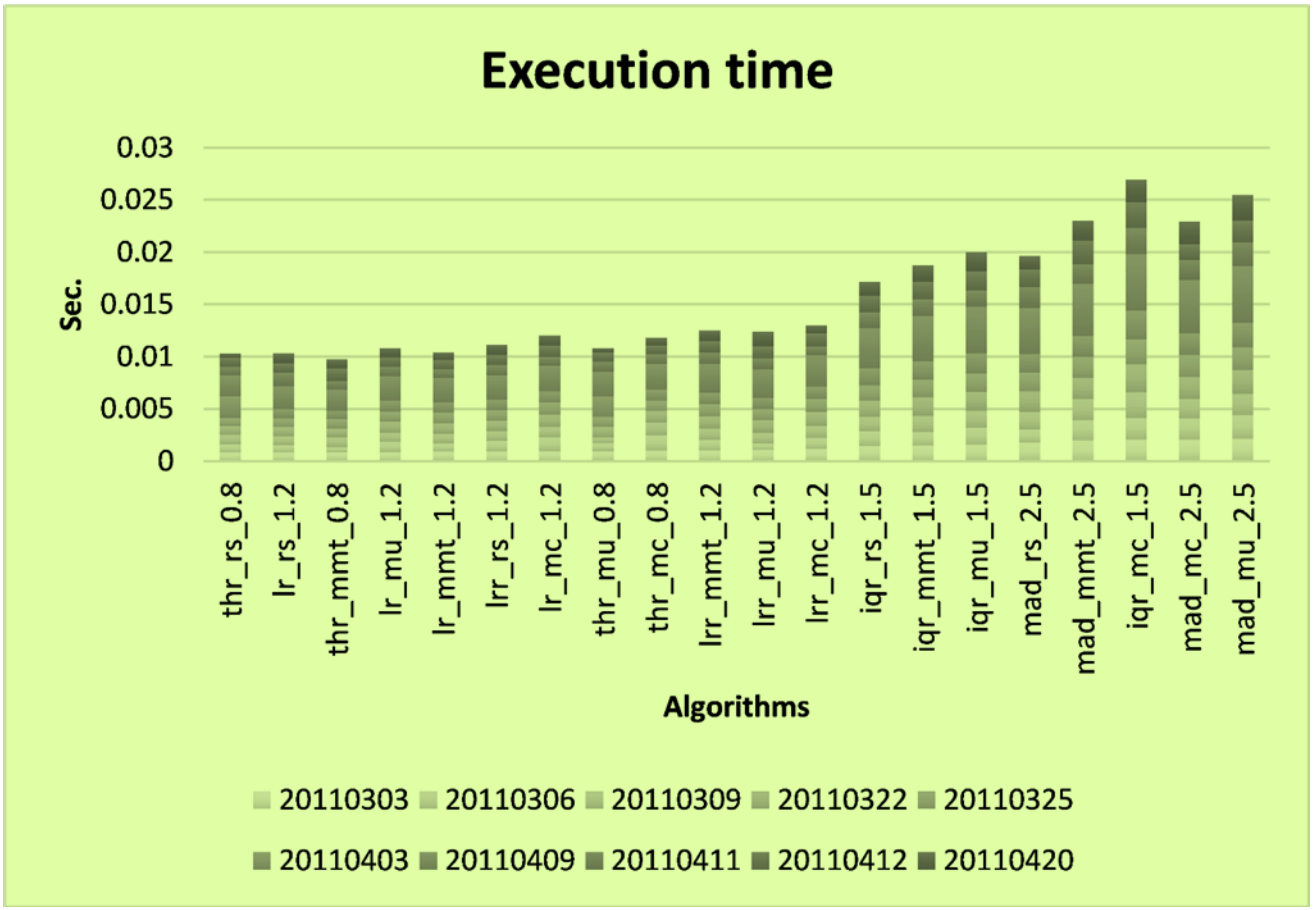

Figure 9: Execution time(in sec) Analysis

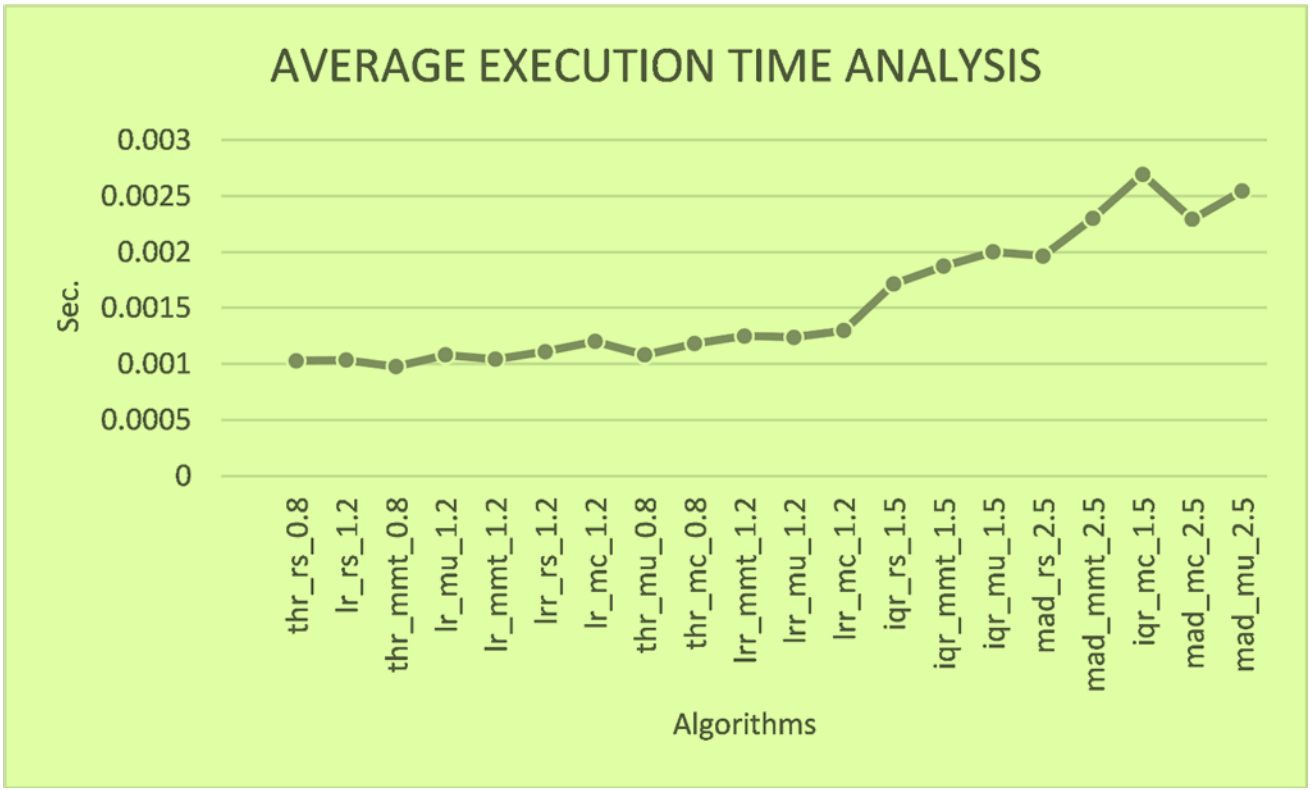

Figure 10: Average Execution time(in sec) Analysis

5 Conclusion and Future Scope

The load balancing scenario was analyzed against the metrics of Energy Consumption, Number of Virtual Machine Migrations, SLA Violation per cent, Meantime before Migration, and total execution time. The combination of robust local regression method with random selection and maximum correlation methods performed the best under energy consumption and number of virtual machine migrations metrics. In the metric for minimal SLA violations, the robust local regression method in combination with minimum migration time and minimum utilization methods performed the best. The minimum migration time method for virtual machine selection in combination with any of the threshold detection methods performs the best for the metric of meantime before migration. Finally for the metric of total execution time, the robust local regression method and static threshold method in combination with random selection and minimum migration time method for VM selection perform the best.

The future scope for these results would lie in choosing the best combination of algorithms for specific application or use case. Let us consider the application of a streaming service, the factors that determine an optimal working of this application are low-latency and high audio/video quality despite the access of millions of users at once. For this scenario, robust local regression method in combination with random selection of VMs would be an optimal combination to deal with an overload scenario as this combination performs the best under total execution time and number of VM Migrations metrics. Similarly, for an E-commerce application where the downtime should be low during the sale season, the combination of Inter-Quartile Range and Maximum Correlation method is an optimal solution for the problem. Additionally, the combination of Static Threshold Detection and Maximum Correlation methods would be a proposed solution for a Search engine application where the results must be indexed well to address the user's search needs.

References

[1] A. K. Singh and J. Kumar, "Secure and energy aware load balancing framework for cloud data centre networks," Electron. Lett., vol. 55, no. 9, pp. 540–541, 2019.

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

[3] K. Li, G. Xu, G. Zhao, Y. Dong, and D. Wang, "Cloud task scheduling based on load balancing ant colony optimization," in Proc. IEEE Chinagrid Conf. (ChinaGrid), Aug. 2011, pp. 3–9

[4] K. Nishant, P. Sharma, V. Krishna, C. Gupta, K. P. Singh, Nitin, and R. Rastogi, "Load balancing of nodes in cloud using ant colony optimization," in Proc. IEEE Comput. Modelling Simulation (UKSim), Mar. 2012, pp. 3–8. doi: 10.1109/UKSim.2012.11.

[5] L. Liu, Z. Qiu, and J. Dong, "A load balancing algorithm for virtual machines scheduling in cloud computing," in Proc. IEEE Modelling, Identificat. Control (ICMIC), Jul. 2017, pp. 471–475.

[6] N. S. Dey and T. Gunasekhar, "A Comprehensive Survey of Load Balancing Strategies Using Hadoop Queue Scheduling and Virtual Machine Migration," in IEEE Access, vol. 7, pp. 92259-92284, 2019, doi: 10.1109/ACCESS.2019.2927076.

[7] P. Humane and J. N. Varshapriya, "Simulation of cloud infrastructure using CloudSim simulator: A practical approach for researchers," 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), 2015, pp. 207-211, doi: 10.1109/ICSTM.2015.7225415.

[8] P. Humane and J. N. Varshapriya, "Simulation of cloud infrastructure using CloudSim simulator: A practical approach for researchers," 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), 2015, pp. 207-211, doi: 10.1109/ICSTM.2015.7225415.

[9] P. R. Theja and S. K. Babu, "An adaptive genetic algorithm based robust QoS oriented green computing scheme for VM consolidation in large scale cloud infrastructures," Indian J. Sci. Technol., vol. 8, no. 27, pp. 1–13, 2015.

[10] P. R. Theja and S. K. Babu, "Evolutionary computing based on QoS oriented energy efficient VM consolidation scheme for large scale cloud data centers," Cybern. Inf. Technol., vol. 16, no. 2, pp. 97–112, 2016.

[11] Q. Huang, F. Gao, R. Wang and Z. Qi, "Power Consumption of Virtual Machine Live Migration in Clouds," 2011 Third International Conference on Communications and Mobile Computing, 2011, pp. 122-125, doi: 10.1109/CMC.2011.62.

[12] Sankar, SyamDath, D.. (2014). Migration of Virtual Machines: A Way to Load Balancing in Cloud Environment. International Journal of Advance Research in Computer Science and Management. 2. 393-395.

[13] Shukla, Praveen Pateriya, R.K.. (2015). I Q R based Approach for Energy Efficient Dynamic VM Consolidation for Green Cloud Data Centers. International Journal of Computer Applications. 123. 28-32. 10.5120/ijca2015905618.

[14] Tziritas, Nikos Loukopoulos, Thanasis Khan, Samee Xu, Cheng-Zhong Zomaya, Albert. (2019). Online Live VM Migration Algorithms to Minimize Total Migration Time and Downtime. 406-417. 10.1109/IPDPS.2019.00051.