SYNTHETIC WORKLOAD GENERATION IN CLOUD

Sudha Pelluri[1], Keerti Bangari[2]

[1]*Dept. of Computer Science & Engg, University College of Engineering(A), Osmania University*
[2]*Dept. of Computer Science & Engg, University College of Engineering(A), Osmania University*

Abstract

User requests, together with arrival timestamp is called Workload. The workload can be either the synthetic or real workload. The synthetic workloads are useful to carry out the controlled experiment. For performance evaluation of complex multitier applications the synthetic workload generation techniques are required such as in Banking , E-Commerce , Business deployed in the cloud computing environments. In each class the application has its own characteristics of workload. The important requirement is that the generated workload called synthetic workload should maintain the same characteristics and behavior of real workload. Techniques to generate synthetic workload are discussed in this paper.

Keywords— *Real trace, Characterization of Workload, Faban, Application Benchmarks.*

----------***----------

1. INTRODUCTION

Cloud computing is a trend in technology in the current decade. Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.

The Public Cloud allows systems and services to be easily accessible to the general public. Public cloud may be less secure because of its openness, e.g., e-mail. The Private Cloud allows systems and services to be accessible within an organization. It offers increased security because of its private nature. The Community Cloud allows systems and services to be accessible by group of organizations. The Hybrid Cloud is mixture of public and private cloud. However, the critical activities are performed using private cloud while the non-critical activities are performed using public cloud. Service Models are the reference models on which the Cloud Computing is based. Software as a service SaaS model allows to use software applications as a service to end users. Platform as a service PaaS provides the runtime environment for applications, development & deployment tools, etc .Infrastructure as a service IaaS provides access to fundamental resources such as physical machines, virtual machines, virtual storage, etc.

For performance evaluation in the cloud we need the experimental platforms, workloads and application benchmarks.[1]The Commercial cloud services are increasingly available, popular, complex, large, and difficult to maintain for the administrator . The demanding area problems are solved by the current cloud computing research. Research topics such as the autonomic systems, optimization, dynamic scalability, fault tolerance, virtual machine scheduling and the releasing, hypervisor resource management, and clouds for rent/cost analysis, all are rely on some form of the workload input. The accuracy of research results can vary considerably based on the slight variations to the input. Trace files are client workloads, and serve as input into the cloud. Understanding and simulating the realistic workload characteristics are imperative for making effective design decisions and adding values to the research results.

Generating the realistic workloads, or trace files, can contribute to innovations in numerous areas of cloud computing. The term "workload" refers to the list of user requests, together with arrival timestamp. The workload can be either the synthetic or real workload. Some of the real traces are not publicly available . Hence, researchers need to generate their own trace files .However most of the authors can use the trace files from the Internet such as the ClarkNet trace or WorldCup trace 98. Synthetic workloads are useful to carry out controlled experiments. For performance evaluation of complex multi-tier applications , synthetic workload generation techniques are required . Such applications are Banking , E-Commerce ,Business to Business deployed in the cloud computing environments.

The goal of this paper is to show that the generated workload has statistical similarities with the realistic workloads. An important objective is to ensure that simulated workload preserves important characteristics and behavior of realistic workload. Synthetic workload can be generated by using the open source load generator like Faban. Initially a real workload is being studied and analyzed for specific characteristics and statistical distribution. The Google trace is publicly available in the Google datacenter and trace version 2 is collected from the cluster data. After analyzing the

characteristics in IBM SPSS the analyzed result will be used in VMware workstation with Faban running on it. This will generate Workload with same characteristics as the real workload.

2. WORKLOAD

Workload is an abstraction of the actual work that your instance or a set of instances are going to perform. we treat a workload as an abstraction, because we intentionally leave out a huge component. It is not possible to generate the entire real time workload. There is a need to understand that , workload can be either the real workload or the synthetic workload.

2.1 Real Workloads

Some of the researchers can generate their own traces, running benchmarks or the real applications in the cloud. However most authors can use the trace files from the Internet such as the ClarkNet trace[2] and WorldCup 98 trace.[3].

1. ClarkNet is a full Internet access provider for the Metro Baltimore-Washington DC area.

   The first log was collected from 00:00:00 August 28, 1995 through 23:59:59 September 3, 1995, a total of 7 days. The second log was collected from 00:00:00 September 4, 1995 through 23:59:59 September 10, 1995, a total of 7 days. In this two week period there were 3,328,587 requests. Timestamps have 1 second resolution.

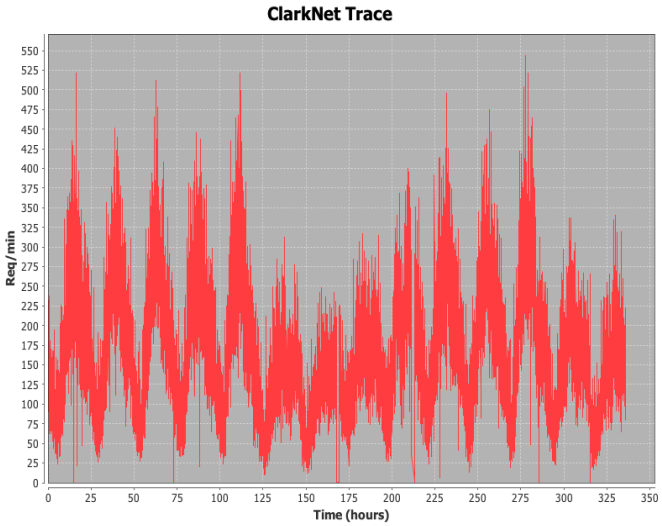


Fig 1: Number of requests per minute for ClarkNet Trace (referred from Auto-scaling Techniques for Elastic Applications in Cloud Environments).

2. WorldCup 98 trace This dataset consists of all the requests made to the 1998 World Cup Web site between the April 30, 1998 and July 26, 1998. And during this period of time the site was received 1,352,804,107 requests.
3. Google cluster data It is also worth mentioning the Google Cluster Data [4], two sets of traces that contain the workloads running on Google compute cells.The first dataset version 1[5], provides traces over a 7 hour period. The workload consists of a set of tasks, where each task runs on a single machine. Tasks consume memory and one or more cores (in fractional units). Each task belongs to a single job; a job may have multiple tasks (e.g., mappers and reducers).The data have been anonymized in several ways: there are no task or job names, just numeric identifiers; timestamps are relative to the start of data collection; the consumption of CPU and memory is obscured using a linear transformation. However, even with these transformations of the data, researchers will be able to do workload characterizations (up to a linear transformation of the true workload) and workload generation .The data are structured as blank separated columns. Each row reports on the execution of a single task during a five minute period.
   a) Time (int) - time in seconds since the start of data collection
   b) JobID (int) - Unique identifier of the job to which this task belongs (may be called ParentID)
   c) TaskID (int) - Unique identifier of the executing task
   d) Job Type (0, 1, 2, 3) - class of job (a categorization of work)
   e) Normalized Task Cores (float) - normalized value of the average number of cores used by the task.
   f) Normalized Task Memory (float) - normalized value of the average memory consumed by the task

4. In second dataset version 2 [6] the trace represents 29 day's worth of cell information from May 2011, on a cluster of about 11k machines. The second trace data has the following contents:
   a) Start and End time: It specifies the start and end time of each tasks.
   b) Jobid and taskindex: The ‘job ID’ and ‘task index’ fields are needed to keep track of the evolution of a given task over time
   c) CPU usage (aka rate) - mean: CPU usage (also known as CPU rate) is measured in units of CPU-core seconds per second: if a task is using two cores all the time, it will be reflected as a usage of 2.0 core-s/s.
   d) Memory usage:Canonical memory usage measurement; the number of user accessible pages, including page cache but excluding some pages marked as stale.
   e) Unmapped page cache memory: Linux page cache (file-backed memory) not mapped into any userspace process;
   f) Page cache memory: total Linux page cache (file-backed memory).
   g) Assigned memory: memory usage based on the memory actually assigned to the container (but not necessarily used).

h) Maximum memory usage: the maximum value of the canonical memory usage measurement observed over the measurement interval. This value is not available for some tasks.

This is the real trace data used in the project first the characteristics and behavior of data is analyzed in the tool called IBM SPSS Stasticts 22.

5. Synthetic Workloads

Synthetic workloads can generated based on the different patterns. According to the Mao and Humphrey [7], there are the four representative workload patterns in cloud environment: Stable, Growing, Cycle/Bursting and On-and-off. Each of them represents the typical application or scenario. A Stable workload is characterized by constant number of requests per minute .The Growing workload pattern may represents the scenario in which a piece of news or video suddenly becomes popular, or consequent Slashdot effect. The Cyclic/Bursting workload may represents the workload pattern of an online retailer, in which the daytime has more workload than the night and the holiday shopping seasons might handle more trace than the normal. The On-and-off workload pattern represent the work to be processed periodically or occasionally, such as batch processing and the data analysis performed everyday .

The synthetic workload is generated similar to the characteristics of real workloads. Having done a detailed analysis and distribution of the workload characteristics of Google workloads using IBM SPSS tool, now the synthetic workloads can be generated.

There is broad range of workload generators, that can be used to generate the simple requests based on any of patterns that are mentioned above or even real HTTP sessions, that mix different actions (e.g. login or browsing) and simulate user thinking times. Examples of workload generators are:

S.No	Workload Generators		
	Name of Workload Generator	**Description**	**Usage**
1	Faban [8]	Faban is a Markov-based workload generator, included in CloudStone stack.	Faban is used in performance, scalability and load testing of almost any type of server application.
2	Apache JMeter [9]	The Apache JMete application is open source software, a 100% pure Java application designed to load test functional behavior and measure performance. It was originally designed for testing Web Applications but has since expanded to other test functions.	Apache JMeter may be used to test performance both on static and dynamic resources (Webservices (SOAP/REST), Web dynamic languages - PHP, Java, ASP.NET, Files, etc. -, Java Objects, Data Bases and Queries, FTP Servers and more).
3	Olio [10]	Olio is a Web 2.0 toolkit to aid in performance evaluations of web technologies.	Olio is primarily for learning Web 2.0 technologies, evaluating the three implementations (PHP, Java EE, and Ruby On Rails (ROR)), to evaluate the performance and scalability of various platforms.
4	VMware Vmmark [11]	VMmark is a benchmark tool used	To measure performance and scalability of applications

Synthetic workloads are suitable for carry out the controlled experimentation. For example we can used to tune the workload in order to test the system under the different number of users or request rates, with the smooth increments or sudden peaks. However, they may not realistic enough, a reason that makes necessary to use the traces from real production systems.

3. APPLICATION BENCHMARKS

Application benchmarks are used to evaluate the server's performance and scalability. Typically, it comprise a web application together with the workload generator that creates the synthetic session-based requests to application. Some commonly used benchmarks for the cloud research are RUBiS , TPC-W and CloudStone.

S.No	Application Benchmark's		
	Name of the Application Benchmark's	**Description**	**Usage**
1	RUBiS[12]	It is a prototype of an auction website modelled after eBay.com.	It offers the core functionality of an auction site (selling, browsing and bidding) and supports three kinds of user sessions: visitor, buyer, and seller. The applications consist of three main components: Apache load balancer server, JBoss application server and MySQL database server.
9	TPC-W[13]	TPC is a non-profit organization founded to define transaction processing and database benchmarks .	It simulates three different profiles: primarily shopping, browsing and web based ordering. The performance metric reported is the number of web interactions processed per second.
10	CloudStone [14]	It is a multi-platform, multi-language performance measurement tool for Web 2.0 and Cloud Computing, developed by the Rad Lab group at the University	The application metric is the number of active users of the social networking application, which drives the throughput or the number of operations per second.

4. IMPLEMENTATION

The implementation is done in two modules.
- **Module 1**: Characterization of real workloads.
- **Module 2**: Generation of synthetic workloads.

In Module 1, the real workload trace file is downloaded, and subjected to statistical analysis. This is done through IBM SPSS stasticts. The workload characteristics of real workload, Google Trace Data are identified and the computations are done using IBM SPSS Stasticts and plots are generated. With these plots or graphs, the distribution of the workload characteristics is found out. As the characteristics and its distribution is available for real workloads, this can be used to generate the synthetic workloads.

In Module 2, an appropriate workload generator tool is used to generate the synthetic workloads. Faban is an open source workload generation tool ,which is suitable in generating the workloads simulating the characteristics of Google workloads. Faban is installed. The Faban consists of a Faban Driver Java File which is programmed appropriately to generate the synthetic workload reflecting the behavior of the real workload. A virtual environment is created using VMware Workstation. Four virtual machines each with CentOS is created. These acts as clients which generate the workloads. Faban is installed in each client machine. Each client machine runs Faban and generate he workload, which is tested in the System Under Test (SUT). For the SUT another virtual machine is created with Ubuntu Server Edition and Apache Web Server. There are different Faban agents in terms of virtual machines in which the workload is generated and tested on System Under Test which runs apache Tomcat Web Server. The apache log file contains the details of all the requests generated, that is the information regarding synthetically generated workloads. This data is used to plot the results using SPSS. The next step is the comparison of the results if real workloads and synthetic workloads and justifying that the synthetic workloads generated has similar characteristics of real workloads.

4.1 Characterization of Real Workloads

In Module 1, the first step is to download workload file, Google Trace data .The trace file containing data is mentioned above .And the real data characteristics is analysed in the tool called IBM SPSS statics.

- SPSS[15]:SPSS Statistics is a software package used for statistical analysis. Long produced by SPSS Inc., it was acquired by IBM in 2009. The current versions (2014) are officially named IBM SPSS Statistics. Companion products in the same family are used for survey authoring and deployment (IBM SPSS Data Collection), data mining (IBM SPSS Modeler), text analytics, and collaboration and deployment (batch and automated scoring services).

First the downloaded trace has to be loaded into the SPSS statistics. The snapshot shows the file loaded into the SPSS.

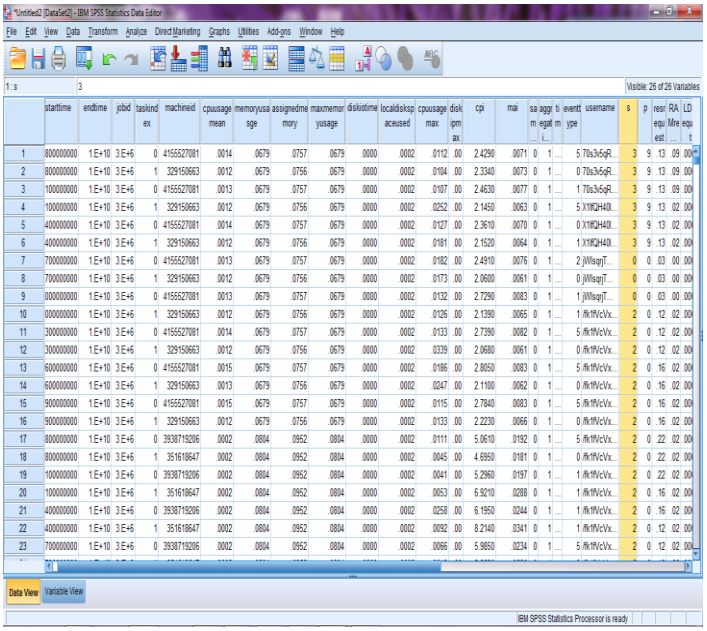


Fig 3 Trace loaded into the SPSS Statists

By using the IBM SPSS statics tool the Google trace data can be sorted per job id and task index. Each task starts at the point 10800000000 and ends at 13200000000.

SPSS RANK can be used to create a variable holding the rank numbers of the values of some other variable.

	starttime	endtime	jobid	Rjobid
1	10800000000	11100000000	3418309	1.000
2	11100000000	11400000000	3418309	1.000
3	11400000000	11700000000	3418309	1.000
4	11700000000	12000000000	3418309	1.000
5	12000000000	12300000000	3418309	1.000
6	12300000000	12600000000	3418309	1.000
7	12600000000	12900000000	3418309	1.000
8	12900000000	13200000000	3418309	1.000
9	10800000000	11100000000	3418309	1.000
10	11100000000	11400000000	3418309	1.000
11	11400000000	11700000000	3418309	1.000
12	11700000000	12000000000	3418309	1.000
13	12000000000	12300000000	3418309	1.000
14	12300000000	12600000000	3418309	1.000
15	12600000000	12900000000	3418309	1.000
16	12900000000	13200000000	3418309	1.000
17	10800000000	11100000000	3418314	2.000
18	11100000000	11400000000	3418314	2.000
19	11400000000	11700000000	3418314	2.000
20	11700000000	12000000000	3418314	2.000

Fig 4 Shows the rank in spss

	starttime	endtime	jobid	Rjobid	taskindex	Rtaskind	PrimaryLast
1	10800000000	11100000000	3418309	1.000	0	1.000	0
2	11100000000	11400000000	3418309	1.000	0	1.000	0
3	11400000000	11700000000	3418309	1.000	0	1.000	0
4	11700000000	12000000000	3418309	1.000	0	1.000	0
5	12000000000	12300000000	3418309	1.000	0	1.000	0
6	12300000000	12600000000	3418309	1.000	0	1.000	0
7	12600000000	12900000000	3418309	1.000	0	1.000	0
8	12900000000	13200000000	3418309	1.000	0	1.000	1
9	10800000000	11100000000	3418309	1.000	1	2.000	0
10	11100000000	11400000000	3418309	1.000	1	2.000	0
11	11400000000	11700000000	3418309	1.000	1	2.000	0
12	11700000000	12000000000	3418309	1.000	1	2.000	0
13	12000000000	12300000000	3418309	1.000	1	2.000	0
14	12300000000	12600000000	3418309	1.000	1	2.000	0
15	12600000000	12900000000	3418309	1.000	1	2.000	0
16	12900000000	13200000000	3418309	1.000	1	2.000	1
17	10800000000	11100000000	3418314	2.000	0	1.000	0
18	11100000000	11400000000	3418314	2.000	0	1.000	0
19	11400000000	11700000000	3418314	2.000	0	1.000	0
20	11700000000	12000000000	3418314	2.000	0	1.000	0

Fig 5 shows the primary last in spss

In the trace file there is a CPU Usage Mean after sorting the data it shows that how much CPU Usage mean is used for each job.

cpuusagemean	cpuusagemean_sum
.001356	.01105100
.001320	.01105100
.001387	.01105100
.001331	.01105100
.001293	.01105100
.001406	.01105100
.001505	.01105100
.001453	.01105100
.001173	.00972200
.001225	.00972200
.001171	.00972200
.001249	.00972200
.001196	.00972200
.001238	.00972200
.001253	.00972200
.001217	.00972200
.000233	.00173180
.000218	.00173180
.000213	.00173180

Fig 6 shows the CPU Mean usage sum

And for the memory usage also shows that how memory used by each job.

memoryusasge	memoryusasge_sum
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.06787000	.54296000
.08044000	.64352000
.08044000	.64352000
.08044000	.64352000

Fig 7 shows the memory usage sum for each job

Here we are finding the utilization percent of CPU usage and RAM request for each user.

username	V2	up
70s3v5qRyCO/1PCdl6fVlnrW8FU/w+5CKRSa72xgclo=	1	.01084800
70s3v5qRyCO/1PCdl6fVlnrW8FU/w+5CKRSa72xgclo=	1	.01056000
X1flQH40lsOBlZKZcc3oourOrsvg/qzZLzBaGgD6lKg=	2	.01056359
jWlsqrjTWzmOBlm1mmGRz6DcxPGiwxxh832jOwL7GTjI=	3	.04259200
jWlsqrjTWzmOBlm1mmGRz6DcxPGiwxxh832jOwL7GTjI=	3	.04137600
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.01216263
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00944166
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00911543
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00938400
70s3v5qRyCO/1PCdl6fVlnrW8FU/w+5CKRSa72xgclo=	1	.00932978
X1flQH40lsOBlZKZcc3oourOrsvg/qzZLzBaGgD6lKg=	2	.00891851
X1flQH40lsOBlZKZcc3oourOrsvg/qzZLzBaGgD6lKg=	2	.03996800
jWlsqrjTWzmOBlm1mmGRz6DcxPGiwxxh832jOwL7GTjI=	3	.01034602
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.01070934
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00786073
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00763488
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00104867
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00098423
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00133438
/fk1fVcVxZ6iM6gHZzqblyq56m5zrmHfpdcZ/zzkq4c=	4	.00176557

Fig 8 shows the utilization percentage

For the first user the CPU Usage Percentage is .01 percentage and for second user so on.

In SPSS by simulating the data we can know that which distribution it follows.

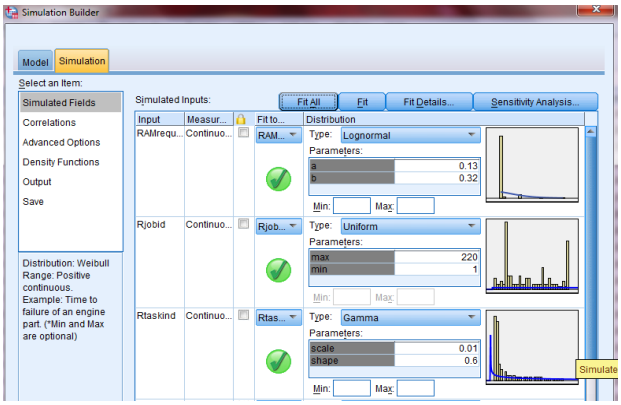


Fig 9 shows the simulation

We know by simulating the data that , the RAM request follows normal distribution. Similarly, we find distributions for other parameters of intrest.

The distribution which are generated in SPSSare useful to enable us to generate workload with similar distribution in faban workload generator.

QQ Plots in SPSS

A q-q plot is a plot of the quantiles of the first data set against the quantiles of the second data set.

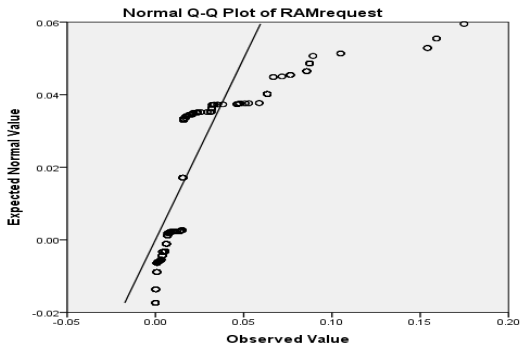


Fig 10 QQ plots for RAM Request

The above snapshot shows the QQplots for RAM request.The RAM request follows normal distribution .Based on the normal distribution the QQplots are generated. In the above snapshot the black small circle shows that how much variance is there in RAM request.

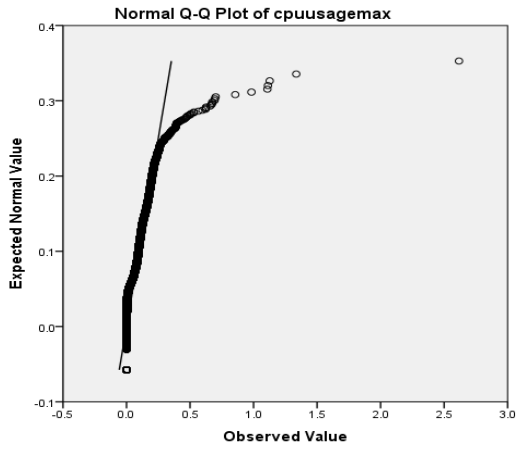


Fig 11 QQplots for CPU usage max

The above snapshot shows the QQplots for CPU usage max.The CPU usage max follows normal distribution .In the above snapshot shows that both the expected and observed values are same with some slight variance at the end in observed value.

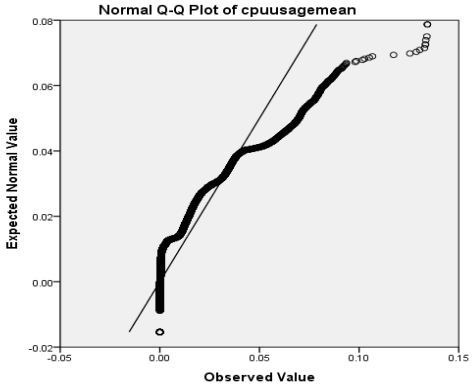


Fig 12 QQ plots for CPU usage mean

The above snapshot shows the QQplots for CPU usage mean. The CPU usage mean follows the normal distribution .In the above snapshot shows that observed value is showing that how much variance it follows.

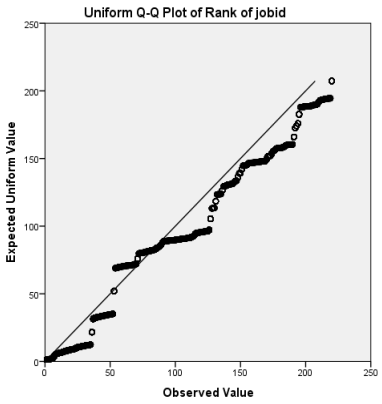


Fig 13 QQ plots for Rank of jobid.

The above snapshot shows the QQplots for jobid.The jobid follows the uniform distribution .

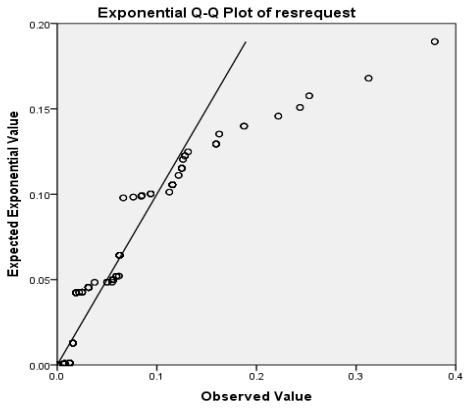
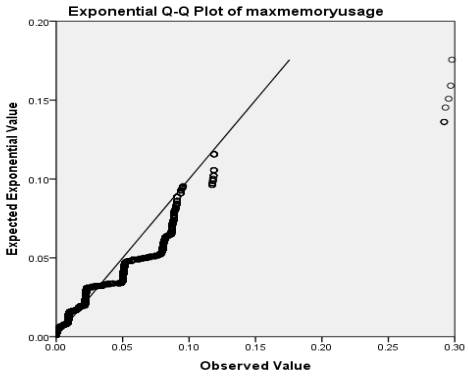


Fig 14 QQ plots for Rank of resrequest.

The above snapshot shows the QQplots for resrequest.The resrequest follows the Exponential distribution



.

Fig 15 QQ plots for Rank of Maxmemory usage.

The above snapshot shows the QQplots for max memory usage .The max memory usage follows the Exponential distribution .

4.2 Generation of Synthetic Workloads

In Module 2, the synthetic workload is generated similar to the characteristics of real workloads. Having done a detailed analysis and distribution of the workload characteristics of Google workloads using SPSS, now the synthetic workloads can be generated. The characteristics and distributions from each of the three categories in the previous section determine the workload design parameters. The properties of the distributions help foster effective workload generator design decisions. It is important to understand the distribution properties prior to generating a workload. The steps involved in the design of synthetic workload generation are:

a) The number of job launches is determined. Within Faban, threads represent job launches,or unique client connection to the server.
b) Establish the number of workload generation machines. Both Task Durations and Tasks per Job determine the number of workload generation machines.
c) The number of job launches per workload generation machine is set. In Faban, threads represent job launches, which are set using the scale parameter. The number of job launches per workload generation machine affects the resulting workload distribution.
d) The appropriate timing or delay parameter is set to define the number of Tasks per Job. In Faban, Tasks per Job cannot be set directly. Instead, define Tasks per Job indirectly using the thinkTime parameter. Increase the think Time parameter to decrease the number of tasks per job.
e) Vary the workload arrival rate as necessary. Client-initiated workloads typically vary and are not flat. The Google trace workload arrival rate fluctuates around a constant average. Faban utilizes a load variation file to vary load patterns.
f) Determine the total test duration, which is the workload runtime. In Faban, the steadyState parameter defines the total test duration.

The task duration of Google workloads can be classified into short duration tasks, medium duration tasks and long duration tasks. Short duration tasks are tasks which has duration in seconds. Medium duration tasks are tasks which has duration in minutes. Long duration tasks are tasks which has duration in hours. Moreover the Tasks per Job is taken as 1 task per job and 100 tasks per job in synthetic workload generation. The workload generation machines are chosen in such a way that each machines generates tasks of a particular duration. That is one for short duration, another for medium duration and third one for hours duration. But as there are different number of tasks per job two workload generating machines are chosen for handling short jobs.One for short jobs with 1 task per job, another for short jobs with 100 tasks per job.

- Creating Virtual Environment and configuration of Virtual Machines.

The virtual computing environment is created using VMware workstation. Utilizing virtual machines consolidates resources and eliminates the need for separate networked computers. Five virtual machines, one server acting as the cloud and four clients simulating hundreds of users and a variety of workloads, are able to communicate within a private network. Utilization of the ping command verifies communication between multiple host machines.

Steps involved are:

a) VMware Workstation 10.0 is installed.
b) Four virtual machines are created with 1 GB RAM and 1 CPU
c) CentOS 7 is installed in each of these four virtual machines. These are the client machines.
d) Another virtual machine is created which acts as the SUT (System Under Test), with 1GB RAM and 1 CPU. This is the server machine.
e) Apache web server is installed and configured in the server machine.
f) Faban is the workload generator tool used in this thesis work. Faban 1.0.1 is downloaded and installed in each of the client machines.

The server and four clients utilized in the experiment have configurations with operating systems, software, and network settings, shown in the table

Table 1 Server Configuration

Parameter	**Description**
Memory	1 GB
Processors	1
HardDisk	2.5 GB
Operating System	Ubuntu Server 14.04.1 LTS
Web Server	Apache2
IP Address	192.168.28.129

Table 2 Client Machine Configuration

Parameter	**Description**
Memory	1 GB
Processors	1
HardDisk	5 GB
Operating System	CentOS 7
Java	Open Java Development Kit 1.7
Load Generator	Faban 1.0.1
IP Address	Faban1 192.168.28.128 Faban2 192.168.28.130 Faban3 192.168.28.131 Faban4 192.168.28.132

- Faban workload generator
  Faban is a Markov-chain-based workload generator, and is widely used for server performance and load testing, also referred to as benchmarking . It contains features that measure and log key performance metrics, and automate statistics collection and reporting. Faban supports numerous servers such as Apache httpd, Sun Java System Web, Portal and Mail Servers, Oracle RDBMS, memcached, and others . It allows the developers to build and modify realistic workloads. Overall, Faban is well documented with manuals, tutorials, blogs, and other web documentation. Due to its distributed and scalable design, Faban is well suited for generating cloud computing workloads . Consequently, Faban is the tool of choice for generating workloads in this research effort.

  Faban is a facility for developing and running benchmarks. Faban supports multi-tier server benchmarks (such as web/cache/database benchmarks) run across dozens of machines. It also supports developing and running a simple micro-benchmark targeting a single component (such as an ftp server.) It has two major components, the Faban Driver Framework and the Faban Harness. Faban Driver Framework is an API-based framework and component model to help develop new benchmarks rapidly . The driver framework controls the lifecycle of the benchmark run as well as the stochastic model used to simulate users. It provides built-in support for many servers such as Apache httpd, Sun Java System Web, Portal and Mail Servers, Oracle RDBMS, memcached etc. It provides a welldocumented interface to add support for any other server. Faban Harness is a tool to automate running of server benchmarks. It also serves as a container to host benchmarks allowing new benchmarks to be deployed in a rapid manner. Faban provides an easy to use web interface to configure and queue runs, and includes extensive functionality to view, compare and graph run outputs. The Faban Harness not only supports benchmarks developed using the Faban Driver Framework but also supports stand-alone benchmarks. Non -Faban benchmarks are supported by defining a benchmark wrapper that identifies the processes involved in preparing and running the benchmark, and reporting results.

In Faban,

a) System Under Test (SUT) comprises all components being tested and varies from benchmark to benchmark. The driver framework and the driver is usually not part of the SUT. Some non-client-server benchmarks can however not differentiate between the SUT and the driver making the driver implicitly part of the SUT.
b) Operation is a single unit of work executed by the user or the driver and timed, and managed by the driver.
c) Operation Mix defines how operations are executed by a simulated user or driver thread.
d) Operation Cycle defines the timing characteristics from one operation to the next.
e) Metric is the resulting data of the operations collected over the steady state time of the benchmark operation.
f) Cycle Time defines the elapsed time between the begin of the previous operation to the begin of the current operation.
g) Think Time defines the elapsed time between the end of the previous operation to the begin of the current operation.

- Starting Faban Master in each client machine
  In each client machine,
  1. The Faban Driver Program is written in JAVA, and build in ANT.
  2. Bring up the Faban harness on the master driver machine using the command# FABAN_HOME/master/bin/startup.sh
  3.The Faban Harness interface is accessed from a browser window.
  4.The browser is pointed to http://<hostname>:9980/deploy and the driver JAVA file is deployed in Faban.
  5. The browser is then pointed to http://<hostname>:9980/
  6. Now Faban is ready to deploy workloads.

Table 2. Description of Tasks in Client Machines

Client Machines	**Tasks**
Machine 1 (faban1)	Seconds Duration,1 Task per Job
Machine 2(faban2)	Seconds Duration, 100 Tasks per Job
Machine 3 (faban3)	Minutes Duration,1 Task per Job
Machine 4(faban4)	Hour Duration,1 Task per Job

- Parameters in Faban in generating workloads
  1. Setting JVM Config JVM heap size limits the size of the workload. The JVM can and will run out of memory. The workload characteristics such as Number of jobs, Number of tasks per job , Number of running tasks affect the amount of memory used within the JVM. J VMs not allocated enough memory to perform the workload throw an error. To correct this issue, increase the amount of memory that the JVM uses by changing the appropriate argument. For example, change the argument - Xmx600m to -Xmx2048m. This will increase the heap size from 600 MB to 2 GB. The operating system and underlying hardware limit the max heap size.

2. Setting RunConfig Here the run parameters in running the workloads are specified.The parameters are
   • Host : The IP address of the client machine is given here
   • Scale : determines the number of threads generated. Here the threads represents the number of jobs generated.
   • Steady State : The total running time of the test.
   - Variable Load : Set to true, for varying load. If there is no load variation it is set to false.
   - Variable Load File : The location of variable load file
3. Setting WebConfig
   - Server Host : Server IP Address is to be entered
   - Port : Port number

- Parameters in each client machine in generating workloads
  - **Machine 1 (Faban1)**

Machine 1 simulates a workload with client connections with, task duration less than one minute, and one task per job.

- **Machine 2 (Faban2)**

Machine 2 simulates a workload with client connections with, short task duration less than one minute, and 100 tasks per job.

- **Machine 3 (Faban3)**

Machine 3 simulates a workload with client connection with, task durations in the 5 to 30 min range, and 1-4 tasks per job.

- **Machine 4 (Faban4)**

Machine 4 simulates a workload with threads representing five client connections; the task durationis in the 1-hour range, and one task per job. The task is not a file download as seen on Machine 3. Itis a series of delays on the web server simulating a long duration task. Large file downloads in theGB range create system instabilities within this small virtual environment, therefore large downloads are avoided.

5. OVERVIEW OF WORK USING FLOW CHART

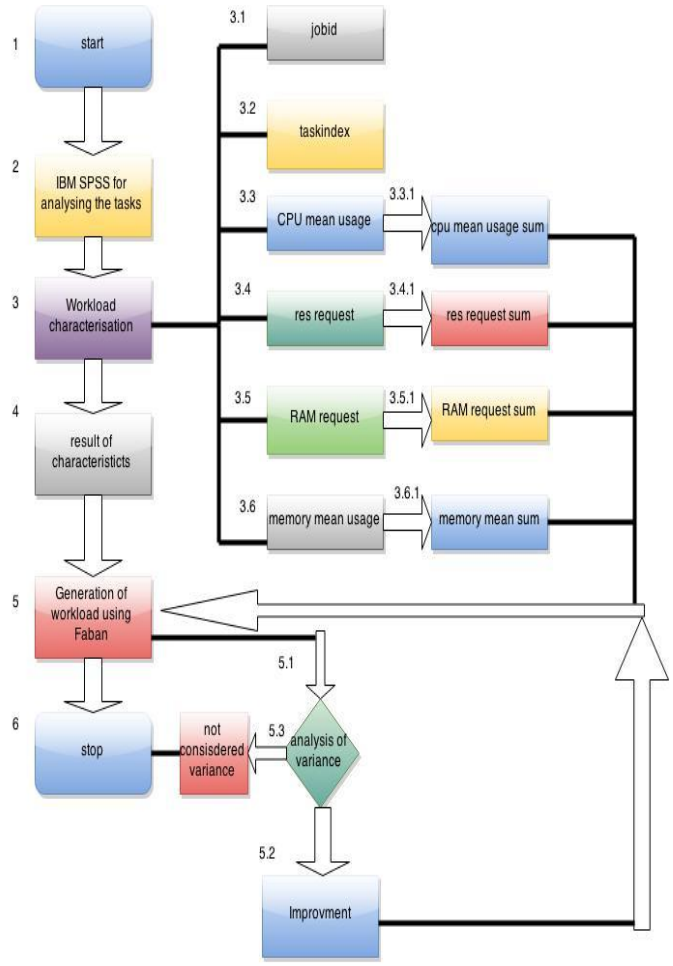


Fig 14 Flow chart of entire set of tasks involved in synthetic workload generation

6. CONCLUSION

In this paper the real workload characteristics are used to generate the synthetic workload such that, the generated workload has similar characteristics and behavior as the real workload. The Google trace version 2 data is the real workload which is publicly available in the Google datacenter. The characteristic of real workload has been analyzed in IBM SPSS. The analyzed result was placed in the VMware workstation with Faban running on it.We have been able to generate synthetic workload which we are going to use in resource provisioning , load balancing energy management and other related research problems being solved in our department.

REFERENCES

[1] Tania Lorido-Botran, Jose Miguel-Alonso, Jose A. Lozano. “Auto-scaling Techniques for Elastic Applications in Cloud Environments”. Technical Report EHU-KAT-IK-09-12.

[2] ClarknetTrace. ftp://ita.ee.lbl.gov/html/contrib/ClarkNet-HTTP.html accessed on 4/5/2014
[3] World Cup 98 Trace (From the Internet Tra_c Archive). http://ita.ee.lbl.gov/ html/contrib/WorldCup.html, 2012. [Online; accessed 13-September-2012].
[4] https://code.google.com/p/googleclusterdata/ accessed on 15/5/2014
[5] https://code.google.com/p/googleclusterdata/wiki/TraceVersion 1
[6] https://code.google.com/p/googleclusterdata/wiki/TraceVersion2
[7] Ming Mao and Marty Humphrey. Auto-scaling to minimize cost and meet application deadlines in cloud workows. In Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis on - SC '11, page 1, New York, New York, USA, November 2011. ACM Press. ISBN 9781450307710. doi: 10.1145/2063384.2063449.URLhttp://dl.acm.org/citation.cfm?id=2063384.2063449.
[8] Faban. faban.org
[9] http://jmeter.apache.org/
[10] https://perfwork.wordpress.com/category/workloads
[11] https://my.vmware.com/web/vmware/info/slug/other/vmware_vmmark/2_x
[12] RUBBoS: Bulletin Board Benchmark. http://jmob.ow2.org/rubbos.html/, 2012. [Online; accessed 18-September-2012].
[13] TPC-W. http://www.tpc.org/tpcw/default.asp, 2012. [Online; accessed 13-September-2012].
[14] CloudStone Project by Rad Lab Group. http://radlab.cs.berkeley.edu/wiki/ Projects/Cloudstone/, 2012. [Online; accessed 13-September-2012].
[15] http://www1.ibm.com/software/in/analytics/spss/products/statistics/
[16] Tania Lorido-Botran, Jose Miguel-Alonso, Jose A. Lozano. "Auto-scaling Techniques for Elastic Applications in Cloud Environments". Technical Report EHU-KAT-IK-09-12.
[17] Arshdeep Bahga, Vijay Krishna Madisetti, " Synthetic Workload Generation for Cloud Computing Applications", Journal of Software Engineering and Applications, 2011, pp 396-410
[18] Lee Gillam, Bin Li, John O'Loughlin, Anuz PranapSingh Tomar, "Fair Benchmarking for Cloud Computing Systems", University of Surrey, March 2012
[19] Raoufehsadat Hashemian, Diwakar Krishnamurthy, Martin Arlitt, "Web Workload Generation Challenges - An Empirical Investigation", HP Laboratories.
[20] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, pp.50–58, 2010.
[21] Pavlos Kranas, Andreas Menychtas, Vasileios Anagnostopoulos, Theodara Varvarigou, "ElasS: An innovative Elasticity as a Service framework for dynamic management across the cloud stack layers". Sixth International Conference on Complex, Intelligent, and Software Intensive Systems 2012, pp 1042-1049
[22] Zhiming Shen, Sethuraman Subbiah, Xiaohui Gu, JohnWilkes, "CloudScale: Elastic Resource Scaling for Multi-Tenant Cloud Systems", In Proc SOCC'11 2011
[23] Sourav Dutta, Sankalp Gera, Akshat Verma, Balaji Vi shwanathan, " SmartScale: automatic Application Scaling in Enterprise Clouds". IEEE Fifth International Conference on Cloud Computing,2012, pp 221-228.
[24] B Urgaonkar, P. shenoy, and et al. " Resource over booking and application profiling in shared hosting platforms." In Proc. OSDI, 2002.
[25] Martin Arlitt, Tai Jin "Workload Characterization of the 1998 World Cup Web Site", Internet Systems and Applications Laboratory HP Laboratories Palo Alto HPL-1999-35(R.1) September, 1999